DOCTORAL THESIS

Extremal Optimisation Applied to Constrained Combinatorial Multi-Objective Optimisation Problems

Gomez-Meneses, Pedro

Award date:
2012

Extremal Optimisation Applied to Constrained Combinatorial Multi-Objective Optimisation Problems

by

Pedro Sebastián Gómez Meneses

BSc(UdeC), BEng(UdeC), PGDip(UdeC)

Submitted in total fulfilment of the requirements of the degree of
Doctor of Philosophy

September 2012

School of Information Technology
Faculty of Business

Typeset in LaTeX2e

Statement of Originality

This thesis is submitted to Bond University in fulfilment of the requirements of the degree of Doctor of Philosophy. This thesis represents my own original work towards this research degree and contains no material which has been previously submitted for a degree or diploma at this University or any other institution, except where due acknowledgement is made.

Pedro Sebastián Gómez Meneses

Date: September 2012

Submitted for examination: February 2012
Approved for the degree of Doctor of Philosophy: September 2012

Abstract

Many real-world optimisation problems, both in the scientific and industrial world, can be classified as constrained combinatorial optimisation problems (CCOPs). Some of the most representative examples of these are: assignment, allocation, scheduling, timetabling, layout, design, routing and distribution problems. These sorts of problems usually have a considerable number of either integer or binary decision variables to which finite and discrete ranges of possible values are assigned. This assignation process has to take into account a set of capacity constraints that must be satisfied (e.g. weight, volume, workload, resources). Additionally, there exists an increasing demand in these sorts of problems to derive solutions that strike a balance between two or more desirable but incompatible objectives, such as maximising component strength while minimising component weight, maximising distance while minimising resource consumption or maximising productivity while minimising runtime. These are commonly referred to as multi-objective problems. Constrained problems, either single or multi-objective, are difficult tasks to be modelled and solved by conventional mathematical techniques. As such, an important area of research is in the domain of novel meta-heuristics that can efficiently solve such problems.

In the last few decades, nature-inspired meta-heuristics have become an effective and efficient alternative used to solve single-objective and multi-objective CCOPs. These meta-heuristics capture ideas and features present in nature or our environment, which are then implemented as search algorithms. These search mechanisms have the ability to explore large and complex search spaces, looking for one or more optimal solutions.

Genetic Algorithms (GAs), Memetic Algorithms (MAs) and Ant Colony Optimisation (ACO) are some of the more representative and successful meta-heuristics used by nature-inspired approaches. The features found in nature represented as an algorithm through these methods generally use a considerable number of operators and parameters that must be properly set. However, these methods are not the only techniques that can be used. Due to the increasing requirement to solve larger and more

complex problems, it is necessary to use more efficient search mechanisms to solve them in a reasonable time. Hence, new competitive approaches, such as Extremal Optimisation (EO), are emerging with simple and straightforward search mechanisms based on a biophysics inspiration.

Extremal optimisation is a meta-heuristic that is relatively little explored compared to other more recognised approaches (such as GAs). Most of the work concerned with extremal optimisation has been applied to single-objective unconstrained optimisation due to the fact that canonical extremal optimisation was originally developed to solve unconstrained problems associated with random magnetic fields in physics. However, there are many open lines of research in extremal optimisation that can be exploited, of which some are addressed in this thesis.

The initial contribution of this thesis is in constraint handling through the development of a mechanism called *differentiated fitness evaluation* that complements EO in order to solve constrained problems. Also, there is a contribution toward hybrid methods, taking ideas from other meta-heuristics and developed from two points of view. First, a *secondary search mechanism* is incorporated to improve the solution convergence. As an initial secondary search mechanism, a local search approach based on a double traverse of the solution representation is proposed. Second, due to the fitness calculation scheme in EO, where the components of the solution are evaluated instead of the entire solution, an *inseparable fitness evaluation* technique for problems that do not provide the necessary information to carry out a separable evaluation for each component, is proposed. Finally, the extension of EO to solve multi-objective optimisation using the *hybrid extremal optimisation framework* proposed in this research, is the last contribution that opens up a promising, yet previously unexplored, research line in EO. The multi-objective hybrid extremal optimisation approach performs a simple and efficient search, complementing an aggregating fitness evaluation function with an extension of the previous proposed local search. This novel local search uses a modified lexicographic ordering scheme to explore the multi-objective constrained combinatorial search space. Thus, this thesis completes a series of interesting and productive challenges.

With these contributions, this research project explores the potential and competitiveness of extremal optimisation to solve single-objective and multi-objective constrained combinatorial optimisation problems through the proposal of a Hybrid Extremal Optimisation (HEO) framework. This turns extremal optimisation into a complete and robust meta-heuristic with the necessary tools to solve a wide range of constrained combinatorial optimisation problems.

The comparative performance of HEO, using the differentiated fitness evaluation scheme as the constraint handling mechanism, is studied through a set of benchmark problems. For single-objective optimisation, the multi-dimensional knapsack, bin packing and generalised assignment problem are used. For multi-objective optimisation, the multi-objective knapsack, multi-objective quadratic assignment and multi-objective job-shop scheduling problem are used. The solution of these benchmark problems extend the applicability of HEO to a series of new problems such as capital budgeting, cargo loading, cutting stock, processor allocation, resource scheduling, vehicle routing, layout design and electronic circuits design. Furthermore, a real-world application in antenna design is investigated.

Results demonstrate the successful operation of the constraint handling mechanism incorporated into the hybrid extremal optimisation framework. HEO shows a competitive performance when it is applied to the single-objective and multi-objective problems used in this research. For single-objective problems, a considerable number of optimal or best-known solutions are achieved and results with a low percentage gap with respect to the optimal are produced (when optimal results are not reached). For multi-objective problems, HEO shows that it is able to expand the surface covered by the best-known approximate Pareto-front set towards their ends, discovering new non-dominated points. However, an exception occurs with the multi-objective job-shop scheduling problem because of the difficulty in defining a separable fitness evaluation for the components of the solution in one of the objectives. This issue will be investigated in more detail in future research.

The extension of extremal optimisation developed in this thesis should lead to increase the diversity and quantity of constrained combinatorial optimisation problems solved by this meta-heuristic.

Acknowledgements

I want to thank the following people and institutions for their support in the achievement of this PhD.

My wife *Emma* and my son *Sebastián* for their unconditional love, support, understanding, forbearance, companionship and happiness. I am especially fortunate in having them with me.

My principal supervisor *Dr. Marcus Randall* for his encouraging advice, assistance, guidance and patience throughout the PhD process. He was always able to clarify my philosophical doubts and put me again on the road to successfully complete this heroic adventure.

My co-supervisor *Dr. Andrew Lewis* for his wise advice and knowledge that always came at the right time.

My colleague and friend *Percy* for making my time in Australia a mostly enjoyable experience through many chats where we try to give a logical explanation to this crazy world in which we live.

School of Information Technology at Bond University, for providing me the facilities to undertake my research, the financial support through the *"Faculty Scholarship"* and to attend various conferences.

Bond University for giving me financial support through the *"Thesis Completion Scholarship"* to write the thesis and the *"Publication Scholarship"* to write a final

research paper.

Universidad Católica de la Santísima Concepción for giving me the opportunity to do this postgraduate degree with my family. But also for supporting me financially by giving me the *"Academic Improvement Scholarship"* to complete my PhD.

To all of them, thanks.

Publications

Publications arising from and included in the thesis

 Part of Chapter 3 reprinted, with permission from:
Gómez-Meneses, P., and Randall, M. Extremal optimisation with a penalty approach for the multidimensional knapsack problem. In *SEAL* (2008), X. Li, M. Kirley, M. Zhang, D. G. Green, V. Ciesielski, H. A. Abbass, Z. Michalewicz, T. Hendtlass, K. Deb, K. C. Tan, J. Branke, and Y. Shi, Eds., vol. 5361 of *Lecture Notes in Computer Science*, Springer, pp. 229–238

 Part of Chapter 3 reprinted, with permission from:
Gómez-Meneses, P., and Randall, M. A hybrid extremal optimisation approach for the bin packing problem. In *ACAL* (2009), K. B. Korb, M. Randall, and T. Hendtlass, Eds., vol. 5865 of *Lecture Notes in Computer Science*, Springer, pp. 242–251

 Part of Chapter 4 reprinted, with permission from:
Gómez-Meneses, P., Randall, M., and Lewis, A. A hybrid multi-objective extremal optimisation approach for multi-objective combinatorial optimisation problems. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010* (Barcelona, Spain, July 2010), pp. 292–299

 Part of Chapter 5 reprinted, with permission from:
Gómez-Meneses, P., Randall, M., and Lewis, A. A multi-objective hybrid extremal optimisation approach applied to rfid antenna design. In *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II*, O. Schütze, C. A. Coello Coello, A.-A. Tantar, E. Tantar, P. Bouvry, P. Del Moral, and P. Legrand, Eds., vol. 175 of *Advances in Intelligent and Soft Computing*. Springer Berlin Heidelberg, 2012, pp. 431–446

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Statement of Problem

Many real-world optimisation tasks in management, engineering and science can be represented as Constrained Combinatorial Optimisation Problems (CCOPs). This sort of optimisation consists of finding the best possible configuration for a considerable number of items, resources or variables that make up the problem. This is done in such a way that the objective function associated with the problem is minimised, or maximised, taking into account a set of requirements that constrain the potential possible solutions. The central characteristic of the constrained component is that not all solutions inside the search space are feasible. The eventual solutions could belong to either the feasible region, which contains the set of solutions that satisfy all constraints, or the infeasible region, which contains the solutions that do not satisfy at least one constraint. As a consequence of this characteristic, and depending on the problem, the feasible search space can be represented by two or more non-contiguous regions. Furthermore, in combinatorial problems, the set of feasible solutions has a finite number of alternatives and generally the items of the solution are indivisibles as in practical situations where activities and resources cannot be segmented into smaller parts.

CCOPs can be found in different practical application areas such as vehicle routing, bin packing, two-dimensional cutting, robotic motion planning, flow-shop scheduling, integrated circuit layout, generalised assignment and network flows [24] which are widely applied to solve problems in the industrial, commercial and service sectors. Thus, the importance of dealing with these kinds of problems lies, in finding novel techniques that are able to solve them in an effective and efficient way so as to bring about significant savings in resources and increases in profits (as an example).

The challenge of solving combinatorial problems lies in the computational complexity of this kind of problem since most of them are classified as $\mathcal{NP}$-hard (Nondeterministic Polynomial-time hard) [125]. This complexity can be understood in terms of the relationship between the search space and the difficulty to find a solution. The search space in combinatorial optimisation problems is discrete and multidimensional; that is, a solution in the search space is represented by a large number of components. The higher the dimensionality, the larger the search space.

To try to find an optimal solution for these problems by using conventional numerical mathematics techniques such as Gradient Ascent and Newton's Method [229], which are used originally in continuous search spaces with two or three variables, is difficult if not impossible. These techniques are designed principally to search optimum values into open spaces without constraints using mathematical mechanisms such as the first and second derivation of the objective function. Among the main drawbacks to this adaptation is as follows: a) not all combinatorial problems can be expressed as a derivable function; b) many combinatorial problems use multi-dimensional decision variables with integers $\mathbb{Z}$, and not real numbers $\mathbb{R}$; c) the search space in combinatorial problems is not continuous; and d) it is necessary to adapt the inherent constraints of combinatorial problems.

Alternatively applied mathematical methods used to solve numerical problems can be adapted to deal with optimisation problems that have a discrete search space; for instance, Linear Programming or Integer Linear Programming [77]. These methods require that both objective function and constraints are expressed in a linear form. The search technique works through an enumerative exploration of the search space to find the optimal solution. This search can be a difficult task as the size of the search space grows exponentially with the dimension of the problem. To reduce the search space to a point where the optimal solution can be located with certainty, there are some techniques that use the concept of cutting or bounding part of this space such as Branch and Bound and Cutting Planes [306]. Besides, traditional artificial intelligence methods based on tree search algorithms or declarative programming paradigms, such as A^* and Constraint Logic Programming [242], have been applied to find optimal solutions; however, these methods present inconveniences when the input size of the problem increases due to the expensive use of memory and runtime [198].

For some combinatorial problems with a moderate instance size (search space) it is possible to design exact algorithms that are able to solve them effectively [304]. However, most combinatorial problems are difficult to solve with exact algorithms, especially when the size of the instances increases, and solely solutions that converge

to the optimum can be found in reasonable time using approximate methods.

In recent decades, disciplines such as operations research and computer science have been developing approximate algorithms with the objective of obtaining good solutions at a reasonable computational cost for these types of problems. More specifically, the emerging fields of soft computing and computational intelligence in computer science have been working in the exploration of new, effective and efficient nature-inspired meta-heuristics which can be applied to solve CCOPs in both the single-objective and multi-objective case. These nature-inspired approaches are an interesting area of research within the approximate methods which explore the search space by imitating some behaviour from nature in order to find optimal or near-optimal solutions. Nature-inspired meta-heuristics can be classified in two main branches, the bio-inspired and physics-inspired techniques. One of the most representative methods from the bio-inspired techniques is Genetic Algorithms (GAs) [128]. GAs are based on the concept of Darwinian species evolution focusing on the natural selection and heredity of genetic material through generations with the objective of improving species. On the other hand, the most classical physics-inspired technique in optimisation is the Simulated Annealing (SA) heuristic. SA is motivated by a metallurgy technique used to improve the quality of materials, which consists of heating the material to be treated and then applying it to a controlled process of cooling to increase the size of its crystals and reduce their defects. In other words, atoms achieve an optimal configuration at each stage of temperature decrease.

Taking into account that both simulated annealing and genetic algorithms have some disadvantages, such as the high number of iterations that may be needed by simulated annealing and the fast initial convergence plus the need of a correct parameterisation for the mutation and crossover operators in genetic algorithms; in recent times new approaches have been developed. This is in response to the concern about exploring new search strategies which are able to find better approximate results in a simpler way, using the least amount of resources and time possible. Among these new techniques are Ant Colony Optimisation [103], Particle Swarm Optimisation [161], Differential Evolution [277], Artificial Immune Systems [86], and Extremal Optimisation [39, 41] which are beginning to be used as new nature-inspired search methods.

Extremal Optimisation (EO) is a relatively recent and emerging nature-inspired heuristic whose search method is especially suitable to solve combinatorial optimisation problems [39]. EO has the particularity that it is based on both biological and physical inspirations. Thus, EO has its foundations in a simple and robust bio-inspired co-evolutionary model which in turn is based on the physics-inspired Self-Organised

Criticality (SOC) [16]. SOC is used to describe the behaviour of dynamic systems that have the capacity to reach a balanced state by themselves until a new event of major magnitude destabilises them. The frequency with which these events occur follows a power law distribution. These systems include the formation of sand piles, the flow of rivers, and the formation of mountain landscapes and coastlines. The co-evolutionary part takes this physical property to modelling the evolution of species as a self-organised critical process where species are influenced by the behaviour of their nearest neighbouring species in the landscape (via extinction events) in order to reach an equilibrium state. Direct application of the concept of SOC to optimisation can be found, for example, in the work of Krink and Thomsen [179], and Lewis and Abramson [189].

To date, only a moderately small amount of research on EO has been developed which has been applied mainly to solve single-objective problems. These works have shown competitive results and then have been compared only with the widely used evolutionary algorithms. In recent years there has appeared the first attempts to extend EO to solve numerical multi-objective problems. Here there is a challenging open area of research where the horizons of EO can be expanded with significant potential opportunities in studying its applicability to CCOPs and its extension to solve multi-objective problems. This is the topic which will be addressed in this thesis.

1.2 Aim and Research Question

This thesis pursues the following aim:

> *To propose a novel and simple nature-inspired framework to solve constrained combinatorial optimisation problems for both single-objective and multi-objective optimisation based on the extremal optimisation heuristic.*

From this aim emerges the following research question which this thesis will seek to answer:

> *Will extremal optimisation be a competitive heuristic to solve single-objective and multi-objective constrained combinatorial optimisation problems?*

To achieve this aim and respond to the research question, it is necessary to direct the efforts in the accomplishment of a number of objectives which are enumerated below:

1. To incorporate a constraint-handling mechanism that allows extremal optimisation to deal with infeasible solutions.

2. To provide a hybrid extremal optimisation framework to solve single-objective constrained combinatorial optimisation problems.

3. To provide a hybrid extremal optimisation framework to solve multi-objective constrained combinatorial optimisation problems.

4. To incorporate an inseparable fitness evaluation technique which allows extremal optimisation to handle problems that do not provide the necessary information to perform a separable evaluation of components.

5. To demonstrate that the proposed hybrid extremal optimisation framework for multi-objective problems is a competitive method to solve a real-world multi-objective constrained combinatorial optimisation problems.

1.3 Methodology

To achieve the objectives of this work an ordered series of tasks will be pursued, and a brief description of the methodology is given here.

First, the state-of-the-art research in the topics of evolutionary computation, constraint-handling, extremal optimisation, and evolutionary multi-objective problems will be reviewed. In this part, the necessary knowledge about these topics will be obtained in issues such as the characteristics and implementations of conventional heuristics in evolutionary computation, the techniques used to handle constraints in evolutionary optimisation algorithms, the features and strengths of extremal optimisation, and the way in which evolutionary multi-objective optimisation operates plus the evaluation measures used for the analysis of the results.

The proposed mechanisms and frameworks to extend extremal optimisation from its canonical definition toward the multi-objective version will be subjected to a comparative study against those well-known heuristics taken from the literature. Benchmark problems and benchmark data in conjunction with specialised metrics for single-objective and multi-objective problems will be used to measure the competitiveness of this proposal.

To improve the competitiveness of the framework in order to find good quality solutions, it is necessary to complement extremal optimisation with the use of an

appropriate hybrid method. In the present work, this hybridisation will be done with the incorporation of two operators.

In Chapters 3 and 4 the first operator is implemented as a secondary search mechanism to improve the convergence and the exploration capacity of extremal optimisation. This secondary search mechanism will be customised according to the problem being solved.

In Chapter 5 the second operator is developed as a inseparable fitness evaluation technique based on the pheromone structure used by ant colony systems [104]. This technique will allow a fitness evaluation for the components of the solution when the problem to be solved does not have the necessary information to assess the positive or negative contribution that each component provides to the solution. This second operator will be applied by the proposed framework to solve a real-world multi-objective problem in the field of the design of RFID antennas. This experiment aims to observe and validate the operation of the proposal when it is used on a real-world problem.

1.4 Contributions

The main contributions arising from this thesis will be as follows:

- *To present a mechanism to handle constraints and infeasible solutions for the extremal optimisation heuristic.* This is a significant contribution since EO, in its canonical form, does not have a constraint-handling mechanism. Also, constraint-handling has attracted much interest in recent times [67, 68, 69].

- *To present an extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for single-objective cases.* EO has been applied successfully only in a limited number of single objective problems such as graph partitioning, graph colouring, max-cutting, and spin glass [36, 44]. Additionally, some exploratory work on other COPs such as the travelling salesman [39, 245], multi-dimensional knapsack [245, 130], maximum satisfiability [206], generalised assignment [243], bin packing [143, 244, 131], and dynamic problems [142] has been undertaken. With this framework proposal the performance of EO applied to other CCOPs will be studied. Also, this framework will explore the use of a secondary search mechanism to improve the results obtained by EO so that this thesis will contribute to hybrid approaches.

- *To present an initial extremal optimisation framework which can be applied to*

solve constrained combinatorial optimisation problems for multi-objective cases. The research on multi-objective optimisation has steadily increased over the past few years [69, 97]. However, it is necessary either to put more effort into looking for new ideas to help improve the extension of existing heuristics or to find completely new heuristics which are able to deliver effective and efficient results. The multi-objective framework proposed herein will contribute a set of results to support future research of multi-objective approaches for EO. Also, this framework will be the base for future exploration of a population-based version of EO, which will contribute to opening a new niche of research and will offer a new alternative to solve multi-objective CCOPs.

- *To present an inseparable fitness evaluation technique for the extremal optimisation heuristic to handle problems that do not provide the necessary information to perform a separable evaluation of components.* This is a significant contribution since EO performs a fitness evaluation based on the contribution of each component of the solution. However, it is not possible to perform this sort of fitness evaluation for all CCOPs. Thus, this new feature allows EO to solve a large range of CCOPs. Note that in the literature, a previous attempt to perform an extension like this to EO has not been found.

- *To present an initial extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for a real-world multi-objective case.* This research will present the study of the suitability of EO to solve a multi-objective design-scenario of RFID antennas [193, 246, 300].

1.5 Thesis Outline

The following provides a brief outline of content, for all subsequent chapters in this thesis.

Chapter 2 reviews the literature to give a formal introduction to the basic concepts to be dealt with in this thesis about nature-inspired computing, extremal optimisation, constraint-handling, and evolutionary multi-objective optimisation. Chapter 3 gives a proposal of how to incorporate constraint-handling in extremal optimisation. Also it presents and analyses a single-objective extremal optimisation framework to solve a set of well-known benchmark problems for combinatorial optimisation. Chapter 4 shows and studies a novel multi-objective extremal optimisation framework to solve

some multi-objective combinatorial problems available in the literature. Chapter 5 describes an implementation of the proposed and tested framework, developed in the previous chapters, applied to solve a real-world CCOP related to the automated design of RFID antennas. Finally, Chapter 6 summarises the work developed in this thesis, gives a statement of its contributions and presents the points for future research work.

Chapter 2

Literature Review

2.1 Introduction

This chapter starts by providing an introduction in relation to optimisation applied to combinatorial problems, especially those with constraints.

The aim of optimisation problems is to find either the maximum or minimum value for a task, activity or resource which is generally represented through an objective function. Most optimisation problems have constraints which describe restrictions that either environmental or external factors exert on some particular aspect of the problem. A mathematically formal representation of optimisation problems can be stated as:

$$\min / \max f(x)$$
$$x \in F \subseteq \mathbb{R}$$

where:

x	is a decision variable that represents a solution,
$f(x)$	is the objective function to be maximised or minimised,
F	is the feasible search space limited by the constraints,
$\mathbb{R}$	is the entire search space.

Optimisation can be seen from many points of view depending on particular characteristics of the problem to be solved. A general categorisation can be achieved by considering aspects such as the number of objectives, the presence or absence of restrictions, the domain of the decision variables, the variability of the decision variables in the time, the form as solutions are searched, and the form as problems are

represented. Figure 2.1 illustrates a general classification for optimisation taking into account the previous aspects.

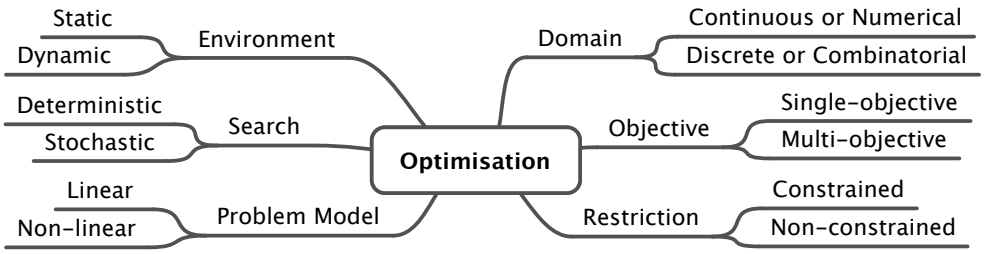


Figure 2.1: General classification for optimisation.

This thesis focuses on constrained combinatorial optimisation with a particular interest in engineering and science because of the many practical problems that can be formulated as combinatorial problems. In these types of problems, the optimum solution must be found within a discrete and finite set of possible solutions. Thus, taking the previous optimisation definition, the domain of the total search space is now restricted to a discrete value set represented by $\mathbb{N}$.

$$x \in F \subseteq \mathbb{N}$$

Combinatorial optimisation problems are habitually simple to formulate but complex to solve [234]. In other words, the time and/or effort necessary to find a mathematical representation for these types of problems should not exceed a normal or reasonable limit. However, the opposite occurs when these problems must be solved because they need a large amount of computational resources. For this reason, many combinatorial problems are categorised as $\mathcal{NP}$-hard so the number of computational steps necessary to obtain an optimal solution increases exponentially, in the worst case, with the number of input variables in the problem.

In the effort to find out novel forms to solve constrained combinatorial optimisation problems, the area of nature-inspired computing has emerged, providing new and auspicious evolutionary meta-heuristics. These original techniques are of special interest for real-world applications as they are able to find near optimal solutions in acceptable run-times.

In the following section, a brief overview about nature-inspired computing meta-heuristics to solve combinatorial optimisation problems will be given. Afterwards, a detailed description of EO as a nature-inspired method to deal with combinatorial optimisation problems will be presented. Next, the most common constraint han-

dling techniques used in evolutionary algorithms to deal with constrained optimisation problems will be presented. Finally, the essential concepts about multi-objective optimisation and a concise description of the two more important evolutionary multi-objective optimisation approaches, together with the quality metrics to measure their performance, will be shown.

2.2 Nature-inspired Optimisation

Nature evolves its organisms slowly over very long time periods to adapt to niche environments. Most of the time, nature has found, in one way or another, a course of action to solve problems which are generally complex due to both the number of components that they have and the obstacles that they must face [94, 259]. Thus, by observing nature's methods, mechanisms or techniques it can be found to develop nature-inspired meta-heuristics. Among the most relevant analogies developed to date can be named the following: evolutionary processes, swarm intelligence, social and cultural behaviours, physical phenomena and human reasoning [87, 105, 116, 247, 309]. In each case, researchers observe their surrounding, looking for ideas that could be modelled and applied, by analogy, to complex optimisation problems using computational algorithms.

A representative case is the homology between ant colony systems [103] and the traveling salesman problem. In both cases, the aim is to travel from point A to point B using the most efficient route. Here, researchers note how ants use a chemical compound called pheromone that is used to mark the best path found at any given time. Subsequently, this ant feature was adapted to an algorithm that has been successfully applied to solve this sort of problem. This and other ideas from nature have been developed to be applied to solve complex problems in engineering, management, and science [13].

Several traditional and emerging nature-inspired meta-heuristics can be found in the literature. These can be classified in three groups according to the natural science field in which the methods are based. Hence, a meta-heuristic can be categorised as bio-inspired, physics-inspired, or biophysics-inspired [32, 94, 214]. Figure 2.2 on the next page illustrates a classification of the most representative meta-heuristics within one of these groups.

Note, this section will only describe the methods that will be referenced in the rest of the thesis. The chosen methods are genetic algorithms, memetic algorithms, ant colony optimisation and extremal optimisation.

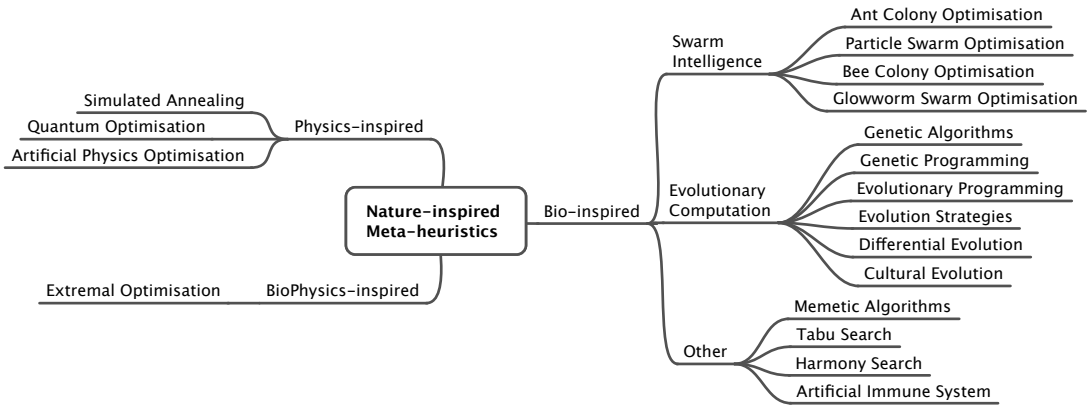


Figure 2.2: Classification of nature-inspired meta-heuristics used for combinatorial optimisation.

The selection of genetic algorithms is based on the benchmark problem sets used to validate the proposed single-objective hybrid extremal optimisation framework in this thesis. Most of the benchmark data obtained from these problem sets were performed using genetic algorithms.

The selection of memetic algorithms is based on the hybrid feature of the proposed framework. From here, some interesting memetic techniques can be identified to be subsequently applied to the enhancement of the extremal optimisation meta-heuristic.

The selection of ant colony optimisation selection is based on the incorporation of some features from this method to complement the proposed multi-objective hybrid extremal optimisation framework when this is applied to solve a real-world problem.

The extremal optimisation meta-heuristic is the main base that supports the principal search mechanism for the proposed framework in this thesis. Despite the fact that there is some research on the use of this technique to solve combinatorial optimisation problems; there has been no comprehensive study to address this meta-heuristic to solve CCOPs, both for the single-objective and multi-objective case. Due to its importance to the framework, this method will be presented in a separate section.

2.2.1 Genetic Algorithms

The genetic algorithms (GAs) meta-heuristic mimics the reproductive process of nature where species evolve over time adapting to their environment [115]. For this, a set of genetic operators are simulated which are then applied to solve search and optimisation problems. According to the work postulated by Darwin in *On the Origin of Species* in 1859 [85], over generations, populations in nature evolve in consonance with

the principles of natural selection. By imitation of this process, genetic algorithms are capable of creating solutions to real world problems. The evolution of these solutions toward optimum values of the problem, depends largely on the proper coding of them. The fundamental concepts of genetic algorithms were provided by Holland [148], and there exists various texts that have good descriptions of them such as Goldberg [128], Davis [89], Michalewicz [208], and Reeves [247].

In nature, species compete for resources for their survival. The most successful individuals will survive and leave more offspring than less adapted or weaker individuals. This can be translated as the better adapted individuals will propagate and will combine their genetic material into future generations. As result of that, there will be super-individuals whose adaptation will be much greater than their ancestors.

Genetic Algorithms, in a direct analogy of this natural behaviour, work with a population of individuals where each individual represents a problem solution which has an associated value or fitness that represents its adaptability degree. Thus, individuals with a high value of adaptability to the problem have a high probability to be selected for reproduction and crossover of its genetic material with other individuals selected in the same form. In such a way, a new population is generated and this will replace the previous one or will merge with the previous one, which gives rise to a new generation with better adapted individuals. That means that in each new generation, these new individuals or solutions will likely explore the most promising areas in the search space converging toward the optimal solution of the problem. Algorithm 1 presents the general GA pseudocode.

Algorithm 1 General GA pseudocode

```
1: Generate an initial population of individuals
2: Evaluate the fitness of each individual in the initial population
3: while the termination criterion is not reached do
4:    for a pre-set population size do
5:       Select a number of individuals according to their fitness for the crossover
         process (the higher the fitness, the higher the probability to be selected)
6:       Apply the crossover operator to the selected individual's chromosomes (par-
         ents) to generate the offspring
7:       Apply the mutation operator to the offspring
8:       Evaluate the fitness of each individual in the offspring
9:       Generate the new population by applying the replacement operator
10:   end for
11: end while
12: return the best solution
```

There exists some aspects that must be taken into consideration when working

with genetic algorithms. Next, the most significant of these will be discussed.

Coding. Each individual (possible solution) is represented through a chromosome which consist of a string of genes (variables or parameters) which in turn store one or more alleles (a specified set of alternatives for each gene). Figure 2.3 gives a graphical presentation of a chromosome and its components.

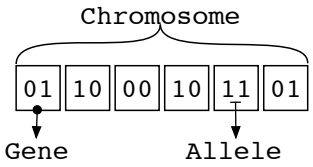


Figure 2.3: Chromosome, gene and allele.

The alphabet used in genetic algorithms is mainly based on a binary code $\{0, 1\}$. When variables have to represent a flip-flop switch; for example, if an item is in or out of a box, a single binary value $\{0, 1\}$ can be used. Alternatively, when variables have to represent an integer or real value, this must be transformed into binary code using some of the coding techniques available in the literature. The correct selection of the coding technique plays an important role in the efficiency of the genetic algorithm process to find optimal solutions [199]. However, this coding process is not mandatory. Also, other representations could be used according to the problem to be solved; for instance, using integer numbers or some specific data structure.

Population. Once the representation of the individuals has been defined, the next step is to determine the size of the population; that is, how many individuals will form the population.

When population size is small, the genetic algorithm does not have enough individuals to generate diversity in future generations. That means the process may converge too quickly toward a homogeneous population which could lead in a detriment of the properly exploration for optimal solutions on the search space. On the other hand, when population size is large the genetic algorithm spends too many computational resources and the time to obtain results may be too much [129].

Hence, the definition of a correct population size is an important aspect for genetic algorithms to develop an appropriate search on the search space. Also, the size of the population plays an influential and crucial role on both the selection process and the population diversity.

The genetic algorithm paradigm originally works with a fixed population size and

there are several studies which analyse varied population size from diverse points of views [32, 128, 129, 196, 241]. However, in recent decades, people have been working with the variable population size approach. Here the population size is varying over time instead of remaining constant.

Fitness. In genetic algorithms, the fitness describes how good the solution is and the fitness function must be formulated in a particular form for each problem to be solved. Once the chromosome structure has been defined, the fitness function takes this chromosome as input and gives a value (generally as, but not limited to, a real number) which defines the adaptability level of the potential solution that is represented in that chromosome.

The fitness value is subsequently used in the selection process. It is fundamental to the carrying out a correct definition of this fitness value because this must represent, in the best possible way, the solution adaptability to achieve an efficient convergence toward optimal solutions.

Selection. In the reproduction process, the selection phase is when individuals from the population are chosen to combine their genetic material to produce offspring with the purpose of gradually improving the adaptability of the next generations.

The parent selection is a stochastic process that favours the best adapted individuals. Each individual is assigned a probability to be chosen which is proportional to its fitness value. Hence, the higher the individual fitness or adaptability, the higher the probability is for it to be selected.

Following this principle, in the literature can be found different points of view of how to carry out this selection. Among these techniques are roulette wheel selection [148], tournament selection [127], ranking selection [18], and the stochastic universal sampling [18].

The most common scheme used is roulette wheel selection (RWS); which can be defined, for a minimisation problem, as follows. Let λ_i be the fitness for an individual i in the population P of size n where n is the number of individuals in the population. Then, the probability p_i that the individual i is selected is given by the following equation.

$$p_i = \frac{\lambda_i}{\Sigma_{j=1}^{n} \lambda_j}$$

Figure 2.4 depicts an illustration of the roulette wheel selection technique.

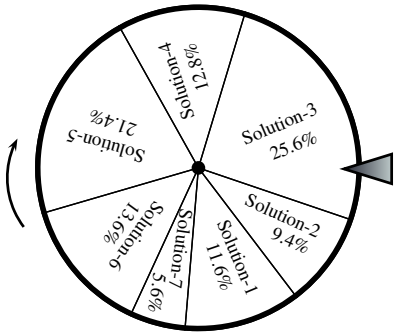


Figure 2.4: Roulette wheel selection.

Crossover. Once parents are selected, their chromosomes are combined using a genetic operator known as crossover. Crossover takes two selected parents and carries out an exchange of genes following a particular scheme such as uniform crossover, one-point crossover, and two-point crossover [1, 13, 89, 116, 128, 175, 199, 208].

For example, one-point crossover cuts the chromosome structure in two parts, generally at one random point. After that, the first part of each parent is joined with the second part from the other parent. As a result of this procedure two new chromosomes are created (referred to as offspring) which inherit genes from both parents.

The crossover operator is applied in a random way; that is, the parent does not always interchange their genetic material and in this case the offspring are a copy of the parent. Figure 2.5 illustrates the one-point crossover operator.

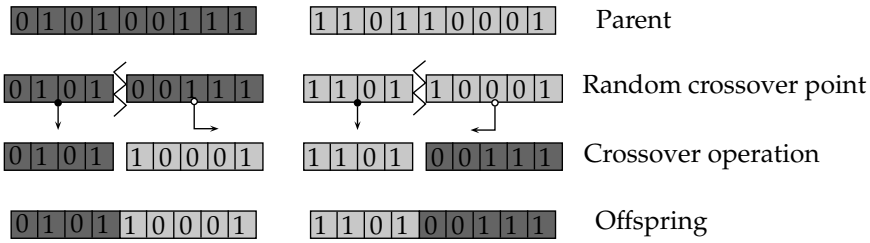


Figure 2.5: One-point crossover operator.

Mutation. The mutation operator is applied to each offspring individually. This operator alters a few randomly selected genes in the chromosome. This alteration occurs at a certain probability p, which is normally small. Mutation complements crossover in the sense that the offspring are able to obtain new genetic material by

way of access to new allele values that the crossover operator cannot reach. This incorporates diversity in future generations (potential solutions) resulting in a wide exploration of the search space.

Mutation depends on the coding used to represent the solution. For instance, when a knapsack problem [202] is to be solved, binary encoding could be used; and in this case, the random mutation scheme can be used. On the other hand, when a travelling salesman problem is to be solved, a permutation representation could be used; and in this case, the exchange mutation scheme can be used. Figure 2.6 illustrates these mutation schemes.

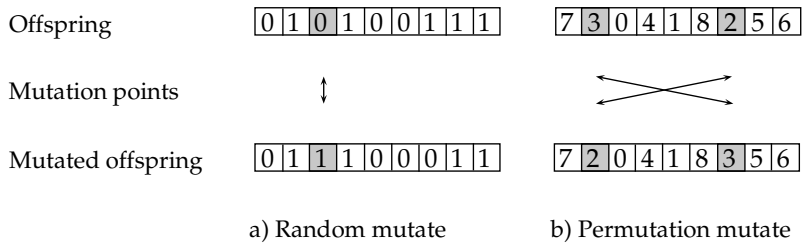


Figure 2.6: Two mutation schemes.

Also, there exists two more mutation schemes that are associated with the encodement used to represent the problem. They are the value encoding mutation and the tree encoding mutation [1, 13, 89, 116, 128, 175, 199, 208]. The former is mainly used when the alleles of the gene are composed of alphanumeric values and this mechanism allows for the modification of genes when the set of values are more complex. The latter is a mechanism suitable for methods such as genetic programming or similar where the chromosome is modelled as a tree of values or objects.

Replacement. Once the reproduction process has ended and the t^{th} population has been generated, the new total population (parents and offspring) must be reduced to the pre-set population size. The population replacement process is generally performed in two different ways. The first is a total replacement of parents by their offspring, which is known as generational replacement [148]. The second is a replacement of a particular number of parents and offspring (selected according to some criterion) until the pre-set population size is complete.

In the latter, parents and offspring have to compete for survival to the next generation and there exists two techniques which are widely used. The first is the steady-state genetic algorithm [283, 302] which replaces an undetermined number n of parents with the same number of offspring, thereby maintaining the population size. The replacement process consists of both an individual selection criterion, such as selecting the

oldest, the worst-fitness, or a random one; and a replacement criterion, such as selecting the best-fitness, or unconditional replacement. The second is referred to as modified genetic algorithm [208] which uses a stochastic mechanism to replace a parent with an offspring. The survival probability for parents and offspring is given by their fitness values that represent their individual adaptability. Thus, parents with low fitness have a high probability of being replaced.

A complementary technique used at this stage of genetic algorithms is elitism, which consists of maintaining the best adapted individual through generations. This technique increases the efficiency to search optimal solutions at the cost of running the risk of decreasing the population diversity.

Termination. The genetic algorithm should stop when it reaches the optimal solution, but as this is usually unknown, the algorithm must define an ending condition to finish the population evolution through generations, which is referred to as termination criterion. In other words, the computational program must stop its iterations in a reasonable time in order to be competitive against other methods. The termination criterion is the component responsible for indicating whether to continue the search for the optimum solution or stop.

Among the termination criteria, the more commonly used are as follows:

- *Generation number.* A pre-set number for the maximum number of generations is defined. If the computational program reaches this maximum number then the evolution stops.
- *Evolution time.* A pre-set time for the maximum duration of evolution is defined. If the computational program reaches this maximum time then the evolution stops.
- *Fitness threshold.* A pre-set fitness threshold is defined. For a maximisation problem, evolution stops when the best fitness reaches a value higher than or equal to the pre-set threshold. For a minimisation problem, evolution stops when the best fitness reaches a value lower than or equal to the pre-set threshold.
- *Population convergence.* Evolution stops when there is no change in the population; that is, all population members converge on a single solution.

Advantages and Disadvantages of Genetic Algorithms

The No Free Lunch Theorem (NFL) [305] says that, in general, all heuristic search techniques are mathematically equivalent. That is, there is no single technique that

outperforms the others in all problems. Therefore, it is necessary to know the strengths and weaknesses of each technique to obtain the best performance of these.

Genetic algorithms have been widely applied since their inception and from those applications the following advantages and disadvantages have been derived.

Advantages:

- It does not depend on analytical knowledge of the objective function to produce, on average, reasonably good solutions.
- It is robust compared with other meta-heuristics because of the significant amount of research and number of applications carried out successfully with GAs.
- It has an intuitive operation that does not require confusing rules, sophisticated mathematical transformations nor intricate search mechanisms.
- It is suitable for problems with highly non-linear and non-derivable objective functions; as also, when the search space is highly complex and / or discontinuous.

Disadvantages:

- It is expensive in computing resources to the effect that, as a stochastic technique, it still requires a large number of evaluations to provide a solution which cannot guarantee that it is optimal.
- It requires special attention to avoid a premature convergence in the population which can lead to being trapped in a local optimum.
- The task of representing and encoding the solution in a chromosome form is not very simple for some problems.
- It is necessary to tune a considerable number of parameters to achieve a solution in an efficient and effective way.

Genetic algorithms have been auspiciously used in the last two decades. However, new and promising approaches have been emerging in recent years that are using simpler mechanisms which prevent many of the disadvantages in genetic algorithms and which could achieve competitive results. One of these approaches is extremal optimisation which presents some similarities with genetic algorithms that will be described later and with which a study to prove its competitiveness against other methods will be carried out.

2.2.2 Memetic Algorithm

The idea of Memetic Algorithms (MAs) was proposed by Moscato [217] in 1989 as the result of the study about a genetic algorithm tendency that began to appear in the late eighties. This study shows that a considerable number of genetic algorithm research works were using some local search mechanism to complement them. From this observation came the idea that supports the memetic algorithm. This idea is based on a combination of techniques and strategies from different meta-heuristics in order to maintain the advantages of each and to obtain an improved method. All this is represented as a set of synergistic processes.

The memetic term comes from the concept of meme that was conceived in 1976 by Dawkins [90]. "Meme" can be defined as the entity that contains a unit of imitation. This unit could be any cultural expression that is transmitted by generations as a mimicking process. The meme is a cultural component that can be represented through a behaviour which survives over time by means of the imitation of it. The meme in memetic algorithms can be compared to the gene in genetic algorithms. Memes are transmitted from brain to brain through imitation as genes are passed from chromosome to chromosome through reproduction. A distinguishing feature between both methods is that genes are an invariable element along the evolutionary process; however, memes have the ability to change as a process of self-improvement.

Some meme examples are: the fashion trend among designers, the predilection toward certain paradigms among scientists, or the imitation behaviour of animals. In other words, meme can be understood as those characteristics that are transmitted through generations and have no relation to biological issues, but that have a relation to beliefs, principles, customs or traditions.

Memetic algorithms are based on three main concepts. First, this method uses a population-based approach where the agents (individuals from the evolutionary algorithms point of view) interact with each other, either in a cooperative or competitive mode. An agent could be represented by one or more individuals which could be associated with one or more specific operators. So, an agent in a memetic algorithm is a more elaborated concept than an individual in an evolutionary algorithm. Second, the hybridisation of different heuristics is used to build a more robust and efficient approach to address a specific problem. The more traditional methods that have been employed are the evolutionary algorithms, the diverse versions of simulated annealing and tabu search. Third, it is necessary to have prior knowledge of the problem to be solved. The amount and the quality of the problem-knowledge achieved is fundamental to achieve the correct selection of the heuristics that could be hybridised and the

operators that are necessary to obtain optimality and speed-up the search process.

Additionally, a good representation of the problem plays an important role in the success of the search mechanism. For this reason, the memetic algorithms give the freedom of choosing any different technique from the traditional coding that has been widely used with the evolutionary algorithms. Thus, a memetic algorithm seeks to integrate those techniques that are considered the best in their respective fields, in a pragmatic way providing an appropriate framework to coordinate, in a single search engine, different helpful heuristics.

To describe memetic algorithms, it is essential to say that these work in direct analogy with social and cultural behaviours. Here, the solution to the problem is represented by one or more populations of individuals or agents that follow some tendencies according to the goal of life or objective of the problem. These tendencies are symbolised by different operators that can be applied to the agents. The most common operators are selection, crossover, mutation and local search, but also any other operator that may be necessary to achieve the optimal solution could be used. Throughout the life of each agent, it could change its likings and preferences about some operators as it is obtaining some benefits from these. If the solution represented by an agent is not improving, it could select a new variant of some operator or choose a completely new operator as a fashionable tendency. The decision is highly influenced by the previous knowledge of the problem. Thus, in memetic algorithms, agents that are improving regularly due to the set of operators chosen according to the different tendencies along the evolutionary process, generally, will obtain a higher value of adaptability or fitness and therefore a higher probability to survive along generations.

Algorithm 2 presents the general MA pseudocode.

Algorithm 2 General MA pseudocode

```
1: Initialise the population
2: Apply the local search operator to the initial population
3: repeat
4:    Apply the selection operator
5:    Apply the crossover operator
6:    Apply the mutation operator
7:    Apply the local search operator
8:    Apply any other operator according to the previous knowledge of the problem
9:    Generate the new population
10:   if Population has converged then
11:      Re-start population
12:   end if
13: until termination condition is satisfied
14: return the best solution
```

Some additional elements will be explained bellow to achieve a more complete understanding of memetic algorithms.

Individual vs agent. The evolution of an optimisation problem is performed by the improvement of its solution toward the optimum value. This solution is characterised by the abstract term known as "the individual". When evolutionary methods are applied to solve any optimisation problem, it is possible for them to work with only one individual or with a group of them gathered in a population. This individual is generally a static structure which along the evolutionary process does not suffer any modification in its configuration, only its state.

Memetic algorithms expand the concept of individual to an agent. Here, an agent could be an individual or a group of them to which could be applied a series of different operators independent of the evolution of the other agents. Also, each agent has a dynamic behaviour in the sense that they could change their configuration in terms of a learning procedure across the evolutionary process.

A further important aspect to be considered is the way in which a solution is encoded. The classical codification used in genetic algorithms is based on schemes, lineal chains and predefined alphabets [199, 285]. However, memetic algorithms allow relaxing the solution representations for the problem by using non-linear forms or any other data structure that may be computationally modelled [217, 218, 221, 220].

Operators. Since memetic algorithms have been derived from evolutionary algorithms, most of the employed operators are the same as those used in the latter (i.e. selection, crossover, mutation, and replacement). However, the distinctive feature of memetic algorithms is that they are able to incorporate additional features beyond classical operators, through the incorporation of relevant information about the problem. Thus, the classical crossover used in genetic algorithms based on a non-guided interchange of genetic material is modified by a guided interchange. Besides, the random introduction of genetic material with the mutation operator in genetic algorithms is customised to a sensible introduction of this genetic material. Also, additional operators could be used. Ideally, operators that have been successfully applied by other heuristics could be merged with traditional operators. Thus, hybrid methods take advantage of the incorporation of these new operators in a synergistic way.

An inherent mechanism widely used with memetic algorithms, to prevent the convergence effect in evolutionary algorithms, is the re-start operator. Thus, when the members of a population became too similar, the re-start operator carries out an ini-

tialisation of the population with the objective of achieving a greater degree of genetic differentiation of the individuals in the population. This allows the new generations to explore new areas in the search space and thus the evolutionary process will not be focused on a small part of the landscape. However, this operator has the drawback of losing all the previous effort to reach that potential optimal solution in the previous neighbourhood. For this reason, recently [221, 219, 220], only a percentage of the population is reinitialised so the potential neighbourhood where the optimal solution could be is kept and the degree of diversity is improved.

Advantages and Disadvantages of Memetic Algorithms

By the fact that memetic algorithms are more elaborate than genetic algorithms, a set of advantages and disadvantages can be found.

Advantages:

- It is specially suitable to solve combinatorial optimisation problems.
- It highlights the best features of the blend of different heuristics in a synergistic way.
- It uses the more robust and flexible concept of an agent than the static representation of an individual in evolutionary algorithms.
- It has performed better than traditional genetic algorithms on combinatorial problems [218, 219, 220].

Disadvantages:

- It is more expensive in computational resources because of the use of additional operators.
- It obtains better results when there exists previous knowledge of the problems which is sometimes hard to obtain.
- It is frequently not so successful for numerical problems because of the extensive use of local search as the additional operator.
- It is sometimes necessary to set more runtime parameters on account of the blend of heuristics.

Memetic Algorithms may be considered as a technique that uses evolutionary concepts in a non-standard way because they do not follow the more traditional evolutionary schemes. However, the incorporation of ideas from a higher level in the field of evolution, such as the social and/or cultural aspects that surround individuals, has given the necessary robustness to consolidate the technique, especially in the field of combinatorial problems. Generally, this incorporation is reflected as a hybridisation of traditional evolutionary algorithms.

Most of the research in memetic algorithms has been developed taking evolutionary algorithms as the base search mechanism. However, according to Moscato [219] it is seen as a healthy sign that the research about the incorporation of other optimisation strategies that use a simpler meta-heuristic which could perform similarly to those using more complex methods as used in evolutionary algorithms. For this reason, this thesis pursues new collaborative strategies in this field through the study of a hybrid framework that uses as its base a novel and simple meta-heuristic.

2.2.3 Ant Colony Optimisation

Ant colony optimisation [102, 103] is a meta-heuristic based on the behaviour of real ant colonies. Ants have the ability to follow the shortest route on their round trip between the colony and a food source. This is possible because ants have access to collective colony-wide information as they move along a chosen path, leaving traces of a substance, called pheromone, behind them. An isolated ant moves essentially in a random way; but in an ant colony, the members or agents tend to follow the pheromone trail left by other ants. Thus, each ant leaves its own pheromone mark along the particular path that it takes. This makes that the path used more frequently is more attractive because it has the pheromone trail more reinforced. However, the pheromone also evaporates over time causing the pheromone trail to become weaker. This decrease in the pheromone level on the less promising paths causes them to be visited more rarely. This process is characterised by a positive reinforcement known as *auto-catalytic behaviour*, where the probability with which an ant chooses a path increases with the number of ants that have previously chosen the same path.

The first ant colony optimisation approach was proposed by Dorigo et al. [102] in 1991. This was applied to solve the traveling salesman problem and the obtained results were quite encouraging. From this initial approach, a robust meta-heuristic, that includes several variants which have been applied to solve diverse optimisation problems, has been developed.

The ant colony optimisation scheme can be seen as a distributed process where

a group of reactive agents interact independently using an indirect communication method to accomplish a common goal. In each iteration of the algorithms, a colony of n ants is created and every ant in the colony builds a solution. This solution is built using a probabilistic mechanism based on the pheromone trail left by the artificial ant plus a-priori information computed heuristically. This is in contrast to other techniques, such as genetic algorithms, that manipulate all solutions over time.

Several meta-heuristics have been proposed based on ant colony optimisation principles. Among the most known of them, that are available to solve $\mathcal{NP}$-hard combinatorial optimisation problems, are Ant System (AS) [102], Ant Colony System (ACS) [104], Max-Min Ant System (MMAS) [278] and Rank-based Ant System (AS-rank) [47]. From these variants, Ant Colony System is one of the most popular of them because of the positive results achieved in combinatorial optimisation, especially with the travelling salesman problem [31, 101].

One of the main features of Ant Colony Systems is its ability of harmonising exploration and exploitation of the search space. Each ant is represented as an agent that has to make a decision every time it faces the necessity of discerning between two or more alternative routes to follow. The different tracks that the ant can take are represented by a graph. Thus, the decision depends on a transition rule that is based on both the level of pheromone deposited on edge (i, j) and a particular heuristic that is associated with the problem to be solved whose value is allocated to edge (i, j). The transition rule allows balance between the exploration of new paths, represented by those edges with both high pheromone levels and favourable heuristic values, and the exploitation of previously accumulated knowledge.

For each ant colony system iteration, every ant performs a movement, either toward the food source or toward the colony. This movement is added to the path that is being traced for each of the n ants. Once ants reach one of the targets, either colony or food source, a cycle is completed and each ant has built a solution.

Equation 2.1 shows the transition rule that assists the choice of the next route to follow during the solution construction process. This allows the k^{th} ant, that is located at node i, to choose node j as the next node in its trajectory. The selection arises as the result of either the exploitation of using the first branch of Equation 2.1 or exploration using the second branch. The parameter q_0 determines the relative importance between exploration and exploitation.

$$j_0 = \begin{cases} \arg\{\max_{j \in N_k(i)}\{\tau_{ij} \cdot \eta_{ij}^{\beta}\}\} & \text{with probability} \quad q_0 \\ J & \text{with probability} \quad (1 - q_0) \end{cases} \tag{2.1}$$

where:

j_0	is the next node to be reached,
$N_k(i)$	is the set of nodes in the neighbourhood of node i for the k^{th} ant,
τ_{ij}	is the pheromone value in the edge ij,
η_{ij}	is the heuristic value in the edge ij,
β	is the weight of the heuristic value,
q_0	is the predefined threshold probability value $0 \leqslant q_0 \leqslant 1$,
J	is a random chosen value with a probability function given by Equation 2.2.

$$S(p_{ij}^k) = \begin{cases} \frac{\tau_{ij} \cdot \eta_{ij}^{\beta}}{\sum_{[l \in N_k(i)]} \tau_{il} \cdot \eta_{il}^{\beta}} & \text{if} \quad j \in N_k(i) \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

where:

p_{ij}^k	is the probability of choosing the next node j from i for the k^{th} ant.
$S()$	is some selection mechanism such as roulette wheel or tournament selection.

In ant colony system there are two levels of updating the pheromone value of each edge, a global update and a local update. The global update is performed after all ants have completed the construction of their solutions. This is applied only to the best solution found so far and only the edges that belong to the solution are able to obtain reinforcement. The pheromone value is updated according to Equation 2.3.

$$\tau_{ij}^t = (1 - \alpha)\tau_{ij}^{t-1} + \alpha\Delta \tag{2.3}$$

where:

τ_{ij}^t	is the pheromone value at time t,
α	is the pheromone decay factor, and
Δ	is the pheromone positive reinforcement for belonging to the best solution.

The local pheromone update has the objective of conserving the diversity among the solutions produced by the ants. The local update is performed when the solution is being built immediately after any ant passes by an edge (i, j). This feedback is

carried out according to Equation 2.4.

$$\tau_{ij}^{t} = (1-\alpha)\tau_{ij}^{t-1} + \alpha\tau_{ij}^{0} \tag{2.4}$$

where:

τ_{ij}^{0} is the initial pheromone value placed on edge ij.

To construct a solution, each ant has a memory which stores the current partial tour. Thus, with this memory the ants are able to determine, in each iteration, the edges that have not yet been visited. Thus, the ant colony system approach can be described in the following general pseudocode illustrated in Algorithm 3.

Algorithm 3 General ACS pseudocode

```
1: Generate the pheromone initialisation
2: while termination criterion is not satisfied do
3:    for each ant do
4:       Allocate randomly one and only one ant in any of the vertices of the graph.
5:    end for
6:    for  each vertices of the graph.  do
7:       for each ant do
8:          Select the next edge in the graph according to Equation 2.1
9:       end for
10:      for each ant do
11:         Apply local pheromone updates according to Equation. 2.4
12:      end for
13:   end for
14:   if a better solution is found then
15:      Set the global best solution with the best solution found in the current iter-
         ation.
16:   end if
17:   Apply local pheromone updates according to Equation. 2.3
18: end while
19: return  the global best solution
```

Advantages and Disadvantages of Ant Colony Optimisation

The ant colony optimisation algorithms have become widely used because of their similarity with many problems in engineering. These problems include, but are not limited to subset problems, routing problems, assignment problems, scheduling problems and constraint satisfaction problems [31, 215]. From this experience, some positive and

negative aspects of this approach have been deduced.
Advantages:

- It is particularly suitable for problems that are represented by a graph with which it is possible to mimic the path search.
- It is an inherently parallel approach, so it can take advantage of concurrent programming and suitable computing hardware.
- It could be used with dynamic problems [142] due to its feature of adapting to changes; for instance, when distances are modified.

Disadvantages:

- It requires more computational resources in memory to keep up the pheromone status and remember the ants' route.
- It is more difficult to model the representation for problems that are different to those about networking or routing.
- It is necessary to pre-set a large number of parameters.

The auto-catalytic effect of the pheromone in ants is a powerful element of communication to indicate the path toward the optimal solution. This concept may be taken and applied as an auxiliary operator on a proposal to develop a hybrid mechanism of optimisation.

2.3 Extremal Optimisation

Nature-inspired methods, like extremal optimisation, have emerged in response to solve those optimisation problems for which conventional mathematical techniques and artificial intelligence methods have difficulties. This can occur because of large and complex forms that search spaces can take and the limitations of current computational systems where the algorithms are performed.

Most combinatorial optimisation problems have been classified as $\mathcal{NP}$-hard and new heuristics like extremal optimisation are giving a new perspective on solving them instead of using the traditional evolutionary algorithms. Extremal Optimisation has the feature of requiring only one algorithm-specific parameter and it uses a minimal amount of memory, which can often lead to a lower computation time. At each iteration, extremal optimisation applies the principle of eliminating one of the weaker or less adapted solution components and replacing it by a random value.

This section will review extremal optimisation as a heuristic for solving constrained combinatorial optimisation problems, having its origins in both bio-inspired and physics-inspired concepts. Thus, a comprehensive summary of its description and current applications will be given.

2.3.1 Description

EO is an evolutionary heuristic proposed by Boettcher and Percus [39] based on the Bak and Sneppen [15] model of co-evolution between species. This model describes species' evolution via extinction events as a self-organised critical (SOC) [16] process. SOC tries to explain the manifestation of complex phenomena in nature such as the formation of sand piles and the occurrences of earthquakes [14]. The main characteristic is that a power law describes the events in the system. Simply put, physical systems have the particularity of having long periods of stability which are occasionally interrupted by spontaneous catastrophic events that trigger a series of critical changes in the system. After this, the system is self-organised to adapt to the unfamiliar surroundings in order to reach a different state of equilibrium. Finally, the system evolves in a new transient period of stability until the next catastrophic change.

The approach of the model proposed by Bak and Sneppen is phylogenetic, i.e. it considers the species as the minimum unit of fitness evaluation instead of the individual. Thus, species are influenced by the fitness of their nearest neighbouring species in the landscape, in order to form the co-evolutionary ecosystem. In nature, self-organising processes can be found which work this way and the natural sciences have already modelled this behaviour [15].

Extremal optimisation has characteristics that distinguishes it from other metaheuristics. For instance, if evolutionary algorithms are taken as a reference point, four differences can be found. First, in extremal optimisation the selection mechanism chooses an element or species with a poor evaluation which does not survive to the next generation. Therefore, this method is based on the elimination of the elements that degrade the system instead of selecting the better elements that survive to the next generation as in evolutionary algorithms. Second, extremal optimisation has the advantage that it requires very few run-time parameters and its implementation is simple. However evolutionary algorithms require tuning a set of parameters and the use of complex genetic operators for proper operation. Third, the fitness value is calculated for each component of the solution; that is, for each species or for each gene of the chromosome. Each component is evaluated according to its contribution the

solution's objective function value. Evolutionary algorithms, however, calculate the fitness for the entire solution or chromosome. Finally, extremal optimisation works with a single solution instead of a population of solutions as in evolutionary algorithms.

Algorithm 4 gives the general EO pseudocode for minimisation problems.

Algorithm 4 General extremal optimisation pseudocode

```
1: Generate an initial random solution X=(x_1, x_2, ..., x_n)
2: Set X_best = X
3: for a pre-set number of iterations do
4:    Evaluate fitness λ_i for each component x_i, 1 ≤ i ≤ n
5:    Choose x_j with the worst fitness for all i
6:    Solution X' in the neighbourhood of X where x_j must change its current value
      to a random value
7:    X = X'
8:    Eva(X) = Evaluate the new solution
9:    if Eva(X) < Eva(X_best) then
10:      Set X_best = X
11:   end if
12: end for
13: return solution X_best and Eva(X_best);
```

The original extremal optimisation algorithm always eliminates the element with the worst fitness value. This action causes the algorithm to have a deterministic behaviour with a high probability of falling into local optima. For this reason, a modification was introduced by Boettcher and Percus [34]. This new algorithm is called τ-Extremal Optimisation (τ-EO). τ-EO[1] improves results and increases the likelihood of escaping from locally optimal solutions simply by establishing a new parameter τ that permits a probabilistic choice of the element to be eliminated rather than necessarily the worst. The modification evaluates and ranks the fitnesses of all elements from the poorest to the best. A rank between 1 and n is assigned to them respectively. Equation 2.5 calculates the probability for each ranked element.

$$P_i = i^{-\tau} \qquad \forall i \quad 1 \leq i \leq n \tag{2.5}$$

where:

- n is the total number of elements evaluated and ranked,
- P_i is the probability that the i^{th} element is chosen.

[1] From hereon in, the term EO is used interchangeably with τ-EO.

Finally, roulette wheel selection (or similar) is then used to choose the element whose value is to change.

Thus, from the general extremal optimisation pseudocode in Algorithm 4 it is necessary to change the instruction number 5 that says:

5: Choose x_j with the worst fitness for all i;

by the following instructions:

5a: Sort fitness λ_i for each x_i from the worst to the best
5b: Generate a random variate $P_{rand} \sim U(1, 0)$
5c: Select an item x_j using the roulette wheel method and probability P_{rand}

2.3.2 Applications of Extremal Optimisation

Extremal optimisation has been applied successfully to some $\mathcal{NP}$-hard combinatorial optimisation problems, though certainly not as extensively as other meta-heuristics, such as simulated annealing, tabu search and ant colony optimisation. This section presents the majority of the extremal optimisation applications developed for discrete optimisation for both single-objective and multi-objective problems.

2.3.2.1 Extremal Optimisation and Single-objective Optimisation

The extremal optimisation heuristic was conceptualised between 1999 and 2000 through the research of Boettcher and Percus [41, 43]. They proposed a new schema of optimisation with few operators and parameters, therefore the algorithm is relatively simple. To demonstrate the efficiency of the proposed method, it was tested on a series of combinatorial problems such as the travelling salesman, graph (bi)partitioning, 3-colouring and maximum satisfiability [39, 42, 45]. Also, it was tested with the physical problem named the spin glass that was modelled as a max-cut problem [43, 35, 36].

Taking the previously mentioned combinatorial and physical problems, extremal optimisation was compared, through a varied number of experiments, with other techniques such as genetic algorithms, simulated annealing and Monte Carlo based on random walks [39, 33, 41, 46, 37, 81, 298, 297]. Subsequently extremal optimisation proved to be equally or competitively superior, depending on the problem, with the techniques previously mentioned.

Exhaustive and detailed research about extremal optimisation, using the graph (bi)partitioning, 3-colouring, and spin glass / max-cut problems, was performed [34,

40, 38, 44]. In these papers, the relationship between the fitness definition and the objective function was studied. In Addition, the effects of different τ values were analysed. The aforementioned analysis was done by taking into account the number of variables of the problem and the runtime.

Some researchers began to use extremal optimisation to solve other related problems in physics which led to the completion of new studies associated with this emerging method. Dall [81] and Onody and De Castro [232] applied extremal optimisation as a alternative method to solve the 3D spin glass problem and so compare it with their proposal. Along the same lines, Middleton [213] proposed a variant of extremal optimisation through a dynamic function that evaluates the fitness to reduce the number of repeated solutions for the 2D and 3D spin glass problems.

Duch, Arena, Danon, and Díaz-Aguilera [82, 107] used extremal optimisation to solve complex network problems from a social point of view. Specifically, they attempted to identify different communities and group them according to some similar characteristics that they share. A similar study, but working with extremal optimisation on a clusterisation model, was developed by Neda, Razvan, Ravasz, Libal, and Gyorgyi [227]. In both cases, the expected results by researchers were fulfilled.

Yom-Tov, Grossman and Inbar [310] introduced a bipartition algorithm using extremal optimisation to classify and analyse a set of signals from the brain when a person is learning a new motor task. This implementation obtained results that, with other proposed algorithms, were not possible. Also, Svenson [282] used extremal optimisation to pre-process signals that come generally in bursts from multiples sensors before sending them to tracking modules in a fusion system. The results for this problem were a reasonably good approximation to the optimal solution.

To achieve better performance with extremal optimisation, some researchers began to develop modifications and extensions of it to solve particular problems. Zhang and Zeng [314] used extremal optimisation as a technique to select an energetically optimal solution from an exponentially large number of protein sequence choices. This approach was used as a stochastic search algorithm applied to computational protein design. The authors used the power law probability distribution in both cases to select the component to be taken out and to select the component to replace it. The results obtained were substantially better than those achieved with simulated annealing and other heuristics attempted to date. Shmygelska [261] applied extremal optimisation to solve the protein folding problem. This work implemented extremal optimisation as the primary search method to explore wide regions of the search space and also used the Monte Carlo technique as the secondary search method to explore local regions

to refine the solution. The results compared favourably with a variant of the Monte Carlo technique.

Iwamatsu [155] joined the conjugate gradient local minimisation method with extremal optimisation. This was applied to solve the conformation optimisation problem associated with Lennard-Jones clusters, which is used to find the lowest energy conformation of atomic clusters. The proposed algorithm was effective in finding the global minimum and also was able to find all the low-lying meta-stable states. Zhou, Bai, Cheng, and Wang [315] also worked on the Lennard-Jones cluster problem with an extension called Continuous Extremal Optimisation (CEO) that consists of two parts: first performs the global search and then the local search. The research results proved to be competitive compared to simulated annealing and genetic algorithms.

Meshoul and Batouche [207] inserted extremal optimisation into an implementation of ant colony systems as a secondary search mechanism to improve the quality of the solutions. This new method was used as a search technique for point-based image registration. Menai and Batouche [205, 206] proposed to change the distribution function used in extremal optimisation from the τ distribution to the Bose-Einstein distribution used in quantum physics. The new version was tested with the maximum satisfiability problem, fulfilling the expected results by researchers.

Furthermore, Sousa and Ramos [269] proposed a new variant of extremal optimisation called Generalised Extremal Optimisation (GEO) which introduced the self-organised criticality concept to the process of the selection of a species to be changed. This proposal was the beginning of a series of works oriented to solve engineering problems in collaboration with other researchers such as Abreu, Galski, Muroaka, and Vlassov [3, 4, 5, 123, 124, 268, 270, 271, 272, 273, 274]. GEO has been applied principally to solve numerical problems where each decision variable is encoded through a set of bits that represent a population of a species. An interesting variant of the GEO algorithm can be found in Sousa et al. [268] where constraint handling is applied to the codification of variables through a flag bit.

Chen, Lu, Yang, and Luo [55, 59, 197] have developed hybrid algorithms combining particle swarm optimisation and genetic algorithms with extremal optimisation that was used as a secondary search mechanism. The hybrid model with particle swarm was tested for numerical problems using six complex unimodal/multimodal benchmark functions. The hybrid model with genetic algorithms was tested on a class of production scheduling problems in manufacturing. The algorithms proposed had successful results in both cases.

There are also a number of studies of the performance properties of extremal opti-

misation. Hoffmann, Heilmann, and Salamon [141, 146] analysed the update process in extremal optimisation as a problem of selecting the next degree of freedom. This study was applied to find the ground state of a system with a complex energy landscape.

Recently a new group of combinatorial problems have been solved using extremal optimisation. Moser and Hendtlass [222, 223] examined the potential of extremal optimisation to solve dynamic combinatorial problems. They used a modified knapsack problem which was adapted to test the extremal optimisation approach in a dynamic environment. Results obtained showed that the extremal optimisation approach demonstrated good behaviour and adaptation when changes in the structure and data of the problem were applied randomly during the problem runtime. The results were compared against an ant colony system solver. Furthermore, the authors proposed an extremal optimisation approach to deal with dynamic aircraft landing [224] which was able to generate feasible solutions that landed all aircraft within their landing ranges with the minimum penalties for early/late landings. The results, compared to the benchmark results available, were highly competitive.

Randall, in collaboration with other researchers, has worked on various extensions of the extremal optimisation model. Randall [243] applied an enhancement extremal optimisation to solve generalised assignment problems. The proposed EO model makes it possible for the method to move between the feasible and infeasible space to search for the optimal value. This proposal is complemented with a partial feasibility restoration operator to reduce the infeasibility of an infeasible solution. The extremal optimisation engine is also combined with a local search operator to improve the quality of a feasible solution. Furthermore, a simple population version of extremal optimisation was developed to compare results with the original single-solution algorithm. Results obtained showed that the model had a better performance in comparison with an ant colony optimisation implementation. Also, Randall and Lewis [245] applied a generalised model of the parallel architecture to extremal optimisation. Parallelism was applied to a population through the concepts of evolutionary population dynamics and self-organised criticality (EPSOC). The model proposed proved to be quite successful when it was applied to a range of real-world problems.

Lewis, Mostaghim, and Randall [191] proposed a unifying meta-heuristic based on population-based optimisation algorithms called evolutionary population dynamics (EDP). This research used ant colony optimisation, extremal optimisation, and particle swarm optimisation as subservient heuristics. Combinatorial and numerical problems for single-objective and multi-objective optimisation were tested. Prelimi-

nary, results showed that this approach was able to find better solutions in relatively small amounts of computational time.

The promising results in the studies of extremal optimisation, that were described above, provide ample motivation to develop more research on it. The application and analysis of this meta-heuristic on unexplored constrained combinatorial problems and its extension to multi-objective optimisation is an interesting and challenging niche to be addressed. The idea to observe the competitiveness and effectiveness of this novel meta-heuristic against the more widely used nature-inspired methods applied to those uninvestigated types of problems, is very stimulating.

2.3.2.2 Extremal Optimisation and Multi-objective Optimisation

Until now, only a small amount of research has been carried out on extremal optimisation applied to multi-objective problems.

Ahmed and Elettreby [8] presented preliminary research on a multi-objective extension of a random variant of the Bak-Sneppen model using a linear aggregating function to solve numerical problems. After that, an equivalent analysis of the aforementioned research was proposed [9] to extremal optimisation using a lexicographic type approach. However, this paper only presented the proposal without any result.

Galski et al. [122, 123, 124] used the generalised extremal optimisation algorithm (GEO) [273] to develop a design for a spacecraft thermal control system. Initially, the GEO was extended to be applied to multi-objective optimisation problems using linear aggregating functions. Afterwards, a new improved version, called M-GEO, was introduced which performs a random lexicographic ordering based on the GEO algorithm. Thus, at each iteration, the algorithm chooses a different objective function to carry out the GEO mechanism. The new solution is processed to verify its Pareto-dominance with the approximated Pareto-front set found so far. However, this proposal is based on the GEO which in turn is based on extremal optimisation and the main use of GEO is to solve numerical problems.

Chen et al. [54, 56, 57, 58] developed an extension of the extremal optimisation heuristic for multi-objective problems based on the Pareto-front ranking concept called multi-objective extremal optimisation (MOEO). They implemented a population-based version of EO called MOPEO. This is the only research where the evaluation of the fitness is founded on the notion of Pareto-dominance. The main characteristic of MOEO is that it works with a mechanism of diversity preservation, an external file to store the approximated Pareto-front set, and a mutation operator. To date, they have only worked on numerical problems, with and without constraints. However, the

drawback of this approach lay in the thoroughgoing fitness evaluation mechanism. The evaluation process is carried out through the mutation of each variable in the solution which generates n new solutions that are ranked according to the Pareto-dominance. This rank value is used to assign the fitness value to each variable in the original solution and then continue with the extremal optimisation scheme. MOEO has been applied to problems with ten and thirty variables only. Although the approach worked successfully for the benchmark problems, the performance for problems with a higher number of variables has not been studied.

Extremal optimisation has been rarely used to solve multi-objective optimisation problems. Considering the promising results obtained with extremal optimisation for single-objective optimisation problems (see Section 2.3.2.1) it is conjectured that the extension of this meta-heuristic to solve multi-objective optimisation problems could generate competitive results. To ascertain this, it is necessary to compare the multi-objective extremal optimisation proposal against the most traditional strategies used in this field. Therefore, in this thesis, the development of a multi-objective framework for extremal optimisation is proposed.

2.4 Constraint Handling

Many problems in engineering are constrained [210], meaning that with a limited number of resources, production requirements of goods or services must be satisfied. The main characteristic of a constrained problem is that not all solutions in the search space are feasible. Solutions either belong to feasible regions which contain the points that satisfy all constraints or infeasible regions that contain the points that do not satisfy at least one constraint. The feasible and infeasible space can be divided into one or more regions depending on the problem [64]. A constrained optimisation problem can be represented mathematically as:

$$\min / \max f(x) \tag{2.6}$$

$$s.t.$$

$$c_i(x) \left[\ <, >, \leqslant, \geqslant, =, \neq \ \right] b_i \tag{2.7}$$

where:

- x is the variable that represents the solution,
- $f(x)$ is the objective function to be maximised or minimised,

$c_i(x)$ is the i^{th} constraint,

b_i is the boundary condition for the i^{th} constraint.

Equation 2.7 defines the feasible search space $\digamma$ which is a subset of the whole search space $\mathbb{R}$ (i.e., $\digamma \subseteq \mathbb{R}$). Figure 2.7 illustrates a generic search space. Here, R is the region where all possible solutions for a particular problem are and F is the subregion or set of subregions in R where solutions that satisfy all constraints are. From the subregions F will emerge the optimal solution.

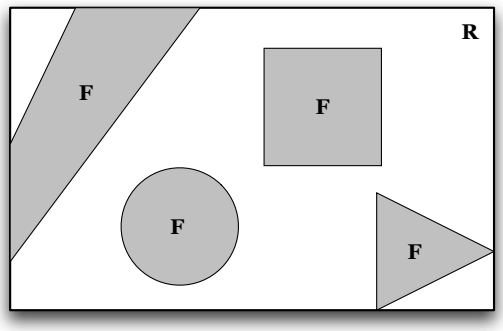


Figure 2.7: Illustrative generic search space.

In mathematics and computational science it is possible to find many techniques to solve constrained combinatorial optimisation problems [178, 184]. From these techniques, this thesis is focused on those that are in the field of nature-inspired metaheuristics. In this field, the constraint handling techniques that are used with evolutionary algorithms are of special interest because of its potential applicability on extremal optimisation. Respecting this, a comprehensive survey on constraint handling techniques in evolutionary algorithms can be found in Coello-Coello [64].

The various constraint handling approaches that have been used to solve constrained optimisation problems can be summarised in the taxonomy proposed by Coello-Coello [64, 66]. This taxonomy is divided into five groups or categories which are: *penalty functions*, *special representations and operators*, *separation of constraints and objectives*, *repair algorithms* and *hybrid methods*. Following, a brief description of each group will be presented.

2.4.1 Penalty Functions

Penalty functions are the most widely used approach in constraint handling [64]. This technique consists of incorporating the evaluation of the constraints into the calcu-

lation of the objective function to impose a penalty on infeasible solutions. That is, penalty functions transform a constrained optimisation problem into an unconstrained optimisation problem by modifying the fitness evaluation of each individual.

The general way to represent the objective function with a penalty approach in evolutionary algorithms is shown in Equation 2.8.

$$f^*(x) = \begin{cases} f(x) & \text{if the solution is feasible} \\ f(x) \pm \sum_{i=1}^{n} p_i c_i(x) & \text{if the solution is infeasible} \end{cases} \tag{2.8}$$

where:

x	is the variable that represents the solution,
$f(x)$	is the objective function to be maximised or minimised,
$f^*(x)$	is the new objective function to be optimised,
$c_i(x)$,	is the i^{th} constraints evaluated for solution x,
p_i	is the i^{th} penalty factor represented by a positive constant.

The choice of the correct value for the penalty factor is an issue not solved yet [88, 250, 251]. As many problems have the optimal solution on the boundary between feasible and infeasible regions [263, 266], it is important to approximate to the boundary with precise movements to reach the optimal. When the penalty factor is assigned high values, the solution switches quickly from an infeasible to a feasible region through coarse movements and after that, when the solution is in the feasible region, it is very difficult for it to approximate to the boundary again. However, if low values are assigned to the penalty factor, then the solution moves through small steps toward the boundary which could take too much time to reach it.

The large number of penalty approaches can be classified as [64, 66]: *death penalty*, *static penalty*, *dynamic penalty*, and *adaptive penalty*. A summary of each is given below.

Death penalty is the simplest and most effective way to manage the restrictions with a very low computational cost [66]. These penalties simply reject any infeasible solution and re-generate a new solution. However, this method is not recommended because it only works for problems that have relatively large feasible regions.

Static penalty is the type of penalty function where the penalty factors are not changed over generations and they stay fixed until the evolutionary algorithm finishes. The penalised function of an infeasible solution for a problem with m

constraints is given by $f^*(x) = f(x) \pm \sum_{i=1}^{m} p_i * \phi_i^k(x)$; where, $f(x)$ is the non-penalised objective function, p_i is the constant penalty factor for the constraint, ϕ_i is a normalised distance of the constraint i applied to the infeasible solution, and k is a user defined exponent whose value is generally 1 or 2 [209]. This method is simple but penalty factors are generally problem-dependent [147, 149, 180].

Dynamic penalty is an extension of the static penalty that works by considering information of the respective generation along with the evolutionary process to define the penalised function. A common practice is to increase the penalty factor over time [158]. According to some researchers [263] dynamic penalties work better than static penalties. For instance, they seemingly work better with arbitrarily constrained optimisation problems. However, defining a good dynamic penalty function is not an easy task [160].

Adaptive penalty works by analysing the results during the evolutionary process which are then used to modify (by increasing or decreasing) the value of some penalty components [74, 75, 230]. Thus, the penalised function shown in Equation 2.8 has to be adapted to become $f^*(x) = f(x) \pm \alpha(t) \sum_{i=1}^{n} c_i(x)$, where the penalty component $\alpha(t)$ increases or decreases its value over time according to some particular state of the population [23, 135]. For instance, measuring the feasibility of the population. The adaptive approaches deal with the penalty component in a more intelligent way than the other penalty categories; however, these approaches may require more work in the definition and configuration of the needed parameters to keep the evolutionary algorithm working properly [266, 265].

2.4.2 Special Representations and Operators

Some particulary difficult problems can be dealt with by changing the original representation of them by another analogous representation. These special representation schemes attack the problem from a different point of view making the search space less hard to explore to find feasible solutions. Because of this change of representation, most of the time is required to redesign some or all of the original operators to work in the same way as in the original representation. A common and simple mechanism used by this technique is to modify the conventional binary coding with another one.

Special representations and operators applied to solve some real-world problems using evolutionary algorithms can be found in the Handbook of Genetic Algorithms [89].

Among the proposals, Decoders [87, 235]is significant as it uses a map (decoder) to make feasible solutions from the chromosomes. Also, Homomorphous Maps (HMs) [176, 177] are the most representative approach of these techniques and they have demonstrated efficient results for problems with convex search spaces. This is illustrated in Figure 2.8. However, the implementation of HMs is elaborate and its run-time is long.

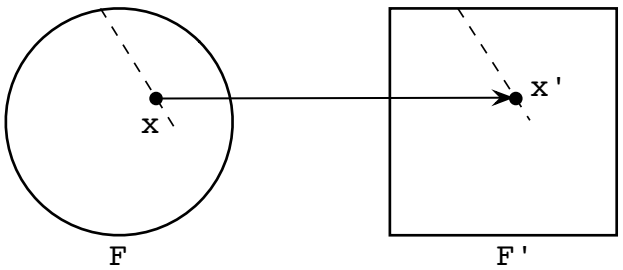


Figure 2.8: Transformation of a convex search space of the original problem F into another search space topologically equivalent F'.

The main disadvantages of this technique are that it is necessary to use extra parameters and the extra computational work associated with the special representations and operators. This technique has shown advantageous results; however, the generalisation to similar problems is not so simple.

2.4.3 Separation of Constraints and Objectives

The separation of constraints and objectives is an approach with a point of view opposite to the penalty function approach. Instead of merging the evaluation of the objective function with the constraints, this technique works with them separately. Some techniques that use this concept are:

Co-evolution. The prey-predator concept from the co-evolutionary model is implemented by Paredis [236] through two populations. One population has constraints which are static in the sense that these do not change over time. However, the fitness value assigned to each constraint might be modified across generations. The other is a conventional population that has possible invalid solutions. The fitness' interaction between both populations has an inverse relation and the value assigned to fitness is the result of a predefined number of encounters between constraints and solutions. On one hand, constraints receive positive feedback when solutions do not satisfy them and negative feedback in

the opposite case. On the other hand, solutions receive positive feedback when constraints are fulfilled and negative feedback in the opposite case.

This approach has presented competitive results but also has had some problems with some extreme cases such as when a constraint is too difficult to satisfy or a solution never fulfils all constraints [237].

Behavioural memory. Schoenauer and Xanthakis [257] proposed a sequential evaluation of the constraints in a similar way to the technique known as *lexicographic ordering* used in multi-objective optimisation. The fitness evaluation is carried out through the following equation:

$$\lambda(x) = C - g_i(x) \tag{2.9}$$

where:

$\lambda(x)$ is the fitness value of a solution,
$g_i(x)$ is the i^{th} constraint evaluated for the solution x,
C is a parameter defined for a particular problem.

The population evolves until a pre-specified percentage of its members become feasible. Then the next constraint is taken to calculate the fitness. The main problem with this approach is that if the constraints are ordered in a different sequence then the results obtained are also different.

Superiority of feasible points. Powell and Skolnick [239], and Deb [96] proposed a technique that applies the concept of giving a better fitness value to feasible solutions and a worse fitness value to infeasible solutions through a ranked order fitness evaluation. The evaluation of the objective function and constraints are separate for feasible and infeasible solutions. A general form to represent the fitness evaluation is shown in Equation 2.10.

$$\lambda(x) = \begin{cases} f(x) & \text{if the solution is feasible} \\ p \pm \sum_{i=1}^{n} c_i(x) & \text{if the solution is infeasible} \end{cases} \tag{2.10}$$

where:

$\lambda(x)$ is the fitness value of solution x,
$f(x)$ is the objective function evaluated for the feasible solution x,
$c_i(x)$ is the i^{th} constraints applied to the infeasible solution x,

p is a parameter defined for a particular problem which separates the evaluation of feasible and infeasible solutions.

In the case of penalty functions, this tends to make the magnitude of the fitness value less reliable. For example, when most of the population violate constraints and the penalty factor is too high then a reduced number of solutions, that satisfy all constraints, will quickly dominate the population and this could lead to a rapid convergence towards a local optimum. On the contrary, if the penalty factor is too low then solutions that violate constraints could dominate the population. This domination will depend on the value of $f(x)$. Thus, with penalty functions, more than one feasible solution might have a fitness value evaluated worse than infeasible solutions. However, with the superiority of feasible points, feasible solutions are never dominated by infeasible solutions and it is not necessary to be worried about the negative consequences when inappropriate definitions of the penalty factors are made.

Although, a few problems of lack of diversity in the offspring population have been reported, for some particular cases; this technique fits with the rank scheme for the selection procedure in extremal optimisation. For this reason, this technique is a potential alternative to be adapted to extremal optimisation as a constraint handling mechanism.

Multi-objective optimisation. Coello-Coello et al. [64] and Deb [97] worked on a technique that proposes to convert a single-objective optimisation problem into a multi-objective problem with $1 + n$ objective functions. The first objective is the original objective function and the remaining n objectives are formed by each one of the constraints. Thus, any multi-objective method can be used to solve this redefined problem. This is a novel and promising technique that researchers such as Hernandez et al. [144], Venkatraman and Yen [294], Zhou et al. [316], Surry and Radcliffe [281] have applied successfully. However, it is important to note that the main drawback of this technique lies in the complexity of applying the multi-objective methods [252].

2.4.4 Repair Algorithms

Liepins [194, 195] developed the first comparative research work showing the benefits of using repair algorithms. Repair algorithms transform infeasible solutions into feasible solutions through a repair operator. For this, infeasible solutions are repaired toward

the nearest feasible solution ensuring the feasibility of the solutions. The restoration process requires tailored heuristic techniques, with the effectiveness of these depending to some extent on programmer skills. The repair process can be applied to many combinatorial optimisation problems. However, there are some problems that this operator may be difficult to implement and computationally costly. Researchers such as Le Riche and Hafka [183], Michalewicz and Nazhiyath [211], Mühlenbein [225], Orvosh and Davis [233], Tate and Smith [284], and Xiao et al. [212, 307, 308] applied this technique successfully to solve numerical problems with nonlinear constraints and combinatorial problems such as the travelling salesman, graph partitioning and path planning and navigation. To date, a small number of approaches to implement a general technique that can be applied to all problems has been developed. The first proposal was presented by Connolly [76] as part of his general purpose simulated annealing approach for 0-1 integer programming problems. This repair algorithm performs a series of switch round (0-1) on the variables of the solution based on the helpful variables to reach the feasibility of the solution. However, the performance of this proposal is proportional to the dimensionality of the problem . Thus, the time necessary to repair infeasible solutions may be large for problems with a high number of variables [2]. On the other hand, there have been other research works that offer a more agile general scheme for some types of problems. For instance, the generalised partial feasibility restoration for assignment type problems proposed by Randall [244] is a simple, non-degenerative, parameter-free process. This process has the aim of complementing a main search method when solutions are in infeasible regions to accelerate their convergence toward feasible regions.

2.4.5 Hybrid Methods

The hybrid methods are built as a combination of the previous-mentioned approaches with other heuristics; for instance, hybrid methods using ant colonies [29, 30, 27, 28, 186, 185, 187], simulated annealing [186], artificial immune systems [136, 137, 311], cultural algorithms [249, 63], and fuzzy logic [182] can be found. Also, these methods can be complemented using some mathematical programming techniques; for example, hybrid methods using the Lagrangian method [6, 163, 162], and simplex method [25]. Thus, each one of these techniques applies its particular way of working to handle constraints.

Hybrid methods are an interesting idea that have been gaining popularity in recent times, with the constraint handling area being no exception. Research conducted so far has obtained promising results showing indubitably the significant potential of

this method when it was applied to solve particular optimisation problems. However, there are also some drawbacks that most of the approaches have had to deal with. For example, the increase in the parameters necessary for their operation or, in most cases, the extra computational effort required for the proper working of these methods. Hybrid methods require more theoretical and empirical investigation to refine and validate the technique. The incorporation of a constraint handling mechanism ,such as the superiority of feasible points technique, to extremal optimisation should be an important contribution in this category.

2.4.6 Constraint Handling for Extremal Optimisation

The canonical extremal optimisation meta-heuristic does not incorporate a formal mechanism to deal with infeasible solutions. For this reason, most of the constrained optimisation problems solved until now have been those where the solution representation is constructed in such a way as to make the generation of infeasible solutions unlikely or indeed impossible. One example is the travelling salesman problem. Here, the salesman has to travel to n different cities visiting each one only once. One way to represent the solution to this problem is by using a vector of length n where each visited city by the salesman is stored in the vector. This representation corresponds to a permutation where an infeasible solution can never be generated.

Unfortunately not all problems can be designed so that their representations do not generate infeasible solutions. One case where this is so is for the assignment problems. Generally, in these types of problems there are some tasks that must be allocated to some agents that have a limited processing capacity. Normally, the number of tasks to be allocated largely exceeds the capacity of the agents and so a selection of them must be made in such a way that none of the agents are overloaded. If this happens then an infeasible solution is generated.

For these problems it is necessary to implement a mechanism to deal with the infeasibility of some solutions that can be generated along the extremal optimisation process. Thus, the incorporation of a constraint handling mechanism to extremal optimisation opens a new horizon of problems that could be solved with this meta-heuristic.

Based on the analysis of the constraint handling mechanisms previously presented, the *Superiority of Feasible Point* approach from the *Separation of Constraints and Objectives* has the set of features that makes it a natural fit to be incorporated into the extremal optimisation meta-heuristic.

2.5 Multi-Objective Evolutionary Optimisation

Many real-world optimisation problems have several optima (maximum or minimum). These optima are equally important, either by having the same numerical value, or because they are not able to establish a criterion to decide which of them is better. Normally, for multi-objective optimisation problems (MOPs), the decision concerning the best answer is given by a human decision-maker.

Multi-objective optimisation can be defined as the challenge of discovering a collection of solutions that satisfy all constraints at the same time and maximise or minimise each one of the objective functions. Generally these objective functions are in conflict with each other in terms of their evaluation. Therefore, a multi-objective optimisation problem must find a solution that optimises all the objective functions for a particular problem.

The Multi-objective optimisation approach resembles single-objective optimisation. The problem is to find a solution vector $\vec{x}$ such that:

$$\min / \max \vec{f}(\vec{x})$$
$$x \in F \subseteq \mathbb{R}$$

where:

$\vec{x}$	is a vector of decision variables $(x_1, ..., x_n)$ that represents a solution,
$\vec{f}(\vec{x})$	is a vector of objective functions $(f_1(\vec{x}), ..., f_k(\vec{x}))$ to be maximised or minimised,
F	is the feasible search space limited by the constraints,
$\mathbb{R}$	is the entire search space.

The more accepted concept of optimality in multi-objective optimisation is the so-called Pareto-optimality. This notion is based on the concept of dominance which is defined below [69].

Definition 1 (Pareto-dominance)**.** *A solution $\vec{s}$ dominates another solution $\vec{s'}$, if and only if, $\vec{s}$ is partially less or great than $\vec{s'}$ and there exists at least an objective where $\vec{s}$ is absolutely less or greater than $\vec{s'}$, denoted by $\vec{s} \prec \vec{s'}$.*

Once the concept of Pareto-dominance has been described, a definition about when a solution is Pareto-optimal is given next.

Definition 2 (Pareto-optimal)**.** *A solution $\vec{s^*}$ is denominated Pareto-optimal, if and only if, there is no other solution $\vec{s}$ which dominates it. That is, $\nexists \vec{s} \in F : \vec{s} \prec \vec{s^*}$.*

After some optimisation process, several Pareto-optimal solutions can be found. The set of results that gathers all of the Pareto-optimal solutions is known as a Pareto-optimal set (P^*).

Definition 3 (Pareto-optimal set)**.** *Pareto-optimal set (P^*) is the set of all Pareto-optimal solution $\vec{s^*}$ such as there is not a solution $\vec{s}$ that $f(\vec{s}) \prec f(\vec{s^*})$.*

The person who chooses the final solutions from the Pareto-optimal set for a particular problem is the decision-maker. The vector that satisfies these definitions is known as the Pareto-front. Thus, for a given multi-objective problem $\vec{f}(\vec{x})$ and a set of Pareto-optimal solutions P^*, the Pareto-front is defined below.

Definition 4 (Pareto-front)**.** *Pareto-front PF^* is the set P^* of all solutions that are Pareto-optimal for a given problem $\vec{f}(\vec{x})$.*

2.5.1 Multi-Objective Evolutionary Optimisation Approaches

Multi-objective techniques can be classified into two types [65]. On one hand, those which do not incorporate the Pareto-optimal concept of the selection operator. On the other, those which perform a rank of the individuals according to if they are dominated or not based on the Pareto-optimal concept.

For the non-based Pareto-ranking methods there are two techniques widely used which are *aggregating functions* [138] and *lexicographic ordering* [119]. The former consists of reducing the m results corresponding to the m objective functions into a single value through weighted summation such as $\lambda(x) = \sum_i^m w_i f(x)_i$, where m is the number of the objectives and w_i is a weight that could be considered to give more importance to some objectives over others. The resulting value $\lambda(x)$ is regarded as the fitness for solution x. The latter is performed by an optimisation process that is applied to one objective separately of the other objectives. Generally, the objectives are ranked according to their importance. Thus, when a single-objective process is carried out on one of the objectives, the results achieved by the other objectives are not affected. This process is repeated for every objective. The use of this technique has been effective until today, especially for those problems where the objectives have different levels of importance.

In the case of the methods based on the Pareto-optimal concept, these have been developed through two generations. The first generation was characterised by the

pioneer approaches which applied the more elemental mechanisms from evolutionary algorithms. With this experience, a second generation emerged that used more complex mechanisms, one of which was the elitism operator that made a substantial difference. The following list shows the most representative approaches since 1984 when the first Pareto-based methods were proposed.

- Vector Evaluated Genetic Algorithm (VEGA) [254, 255, 256] proposed by Schaffer in 1984-1985.
- Multi-Objective Genetic Algorithm (MOGA) [117] proposed by Fonseca and Fleming in 1993.
- Niched-Pareto Genetic Algorithm (NPGA) [150, 151] proposed by Jeffrey Horn et al. in 1993-1994 and Niched-Pareto Genetic Algorithm 2 (NPGA2) [111] proposed by Erickson et al. in 2001.
- Non-dominated Sorting Genetic Algorithm (NSGA) [275] proposed by Snirivas and Deb in 1994 and Non-dominated Sorting Genetic Algorithm II (NSGA-II) [99, 100] proposed by Deb et al. in 2000.
- Strength Pareto Evolutionary Algorithm (SPEA) [324] proposed by Zitzler and Thiele in 1999 and Strength Pareto Evolutionary Algorithm 2 (SPEA2) [322] proposed by Zitzler et al. in 2001.
- Pareto Archived Evolution Strategy (PAES) [169, 170] proposed by Knowles and Corne in 1999-2000.
- Pareto Envelope-based Selection Algorithm (PESA) [79] proposed by Corne et al. in 2000 and Pareto Envelope-based Selection Algorithm II (PESA-II) [78] proposed by Corne et al. in 2001.
- Multi-Objective Messy Genetic Algorithm (MOMGA) [290, 292] proposed by Van Veldhuizen and Lamont in 1999, Multi-Objective Messy Genetic Algorithm II (MOMGA-II) [326, 325] proposed by Zydallis et al. in 2001 and Multi-Objective Messy Genetic Algorithm IIa (MOMGA-IIa) [91, 92, 93] proposed by Day et al. in 2004-2005.
- Micro Genetic Algorithm (microGA) [70, 71] proposed by Coello and Toscano-Pulido in 2001 and Micro Genetic Algorithm 2 (μGA^2) [286] proposed by Toscano-Pulido and Coello in 2003.

From the above list, the most representative and well studied methods are NSGA-II and SPEA2 [69]. Both have been widely applied to many multi-objective optimisation problems which have achieved successful results. For this reason, they have become a de-facto reference method to compare new approaches in this area.

In this research, NSGA-II and SPEA2 will be used to analyse and compare the multi-objective extremal optimisation approach proposed in this thesis. A concise description of them is presented below.

2.5.1.1 NSGA-II

Deb et al. proposed the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [99, 100] with the objective of finding a set of efficient solutions under the concept of Pareto-dominance. This approach has proven to be one of the most fast, effective and robust techniques to solve multi-objective optimisation problems [65]. Also it has been one of the most popular methods to compare new multi-objective optimisers.

NSGA-II is an algorithm based on the original implementation of NSGA. The enhanced approach incorporates the concepts of elitism to improve the convergence and crowding distance to maintain diversity in the Pareto-frontier using a population of fixed size. The composition of the current population is based on a combination of the new and the previous generation. To do this, NSGA-II uses the usual genetic operators for selection and mutation to generate an offspring population from the parent population. Both populations (parents and offspring) are linked together and then the dominance ranking is applied to classify the coupled population in a number of levels, so that the first level is the best level of the population.

A new population is created based on the rank order of the levels; thus, the individuals from the best level are selected first. If the number of individuals in the new population is smaller than the fixed size of the population then the next level is taken into account. On the other hand, if the new added level exceeds the number of individuals allowed in the new population then a truncation operator is applied to that level.

This operator is based on crowding distance which is calculated by the summation of all distances between one individual and its nearest neighbours for each dimension in the space of the objective functions. Thus, individuals with less crowding distance are removed. As a result, a population of descendants of the same size as the initial population is created.

Figure 2.9 shows a scheme of the NSGA-II mechanism. P_t represents the parents population and O_t represents the offspring population at generation t. F_1 represents

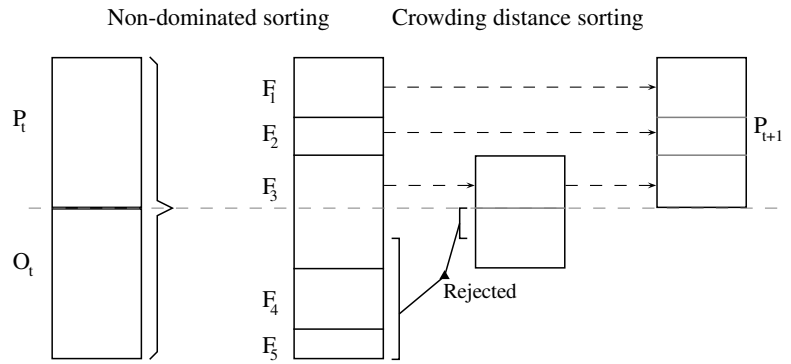


Figure 2.9: NSGA-II

the best mixed approximated Pareto-front set coming from the parents and offspring populations. F_2 represents the second best mixed approximated Pareto-front set and so on.

NSGA-II has one of the best evaluated performances among multi-objective techniques available to date. For this reason, there exists a considerable amount of theoretical research works solving the classical benchmark problems and real world applications solving problems of design, scheduling, management, distribution and classification, among others [69]. However, this technique presents some difficulties when the number of objectives is increased because of the implementation complexity in the crowding distance mechanism.

The general pseudocode of the NSGA-II approach is presented in Algorithm 5.

Algorithm 5 Pseudocode of the NSGA-II.

1: Generate the initial random population P_{ini} of size n
2: Evaluate and rank each individual from population P_{ini} according to the non-domination and the crowding distance criteria
3: **repeat**
4: Generate the offspring population O_t of size n with the standard genetic algorithm operators
5: Evaluate and rank each individual from both population P_t and O_t according to the non-domination (front levels) and the crowding distance criteria
6: Select the best front levels for the new population P_{t+1} filling the new population with the remaining individuals from the last level selected up to complete size n.
7: **until** the stop criteria is satisfied
8: **return** the first non-dominated front

2.5.1.2 SPEA2

Zitzler et al. proposed the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [322]. This approach has been compared successfully against other multi-objective evolutionary algorithms in terms of convergence and diversity on well known benchmark problems [69]. Hence, it has become a reference meta-heuristic for new approaches.

SPEA2 is an algorithm based on the original implementation of SPEA. The new version improves the fitness assignment scheme, the density estimation technique and incorporates a new archive truncation method.

In SPEA2, the initial population is generated using a uniform distribution, the representation of the solutions is binary encoded and the evolutionary operators are standard, such as binary tournament selection, crossing point, and bit-flip mutation.

The real power of SPEA2 lies in the use of elitism. Thus, SPEA2 makes use of an external file A to store the approximated Pareto-front set found so far, which is used with the current population P to form the offspring O for the next generation. Thereafter, a new file A is regenerated through updating the approximated Pareto-front set from the population P and the offspring O.

The file size is fixed and this is often set as the same size as the population. Therefore, there are two special situations where solutions are passed to the file. If the number of non-dominated solutions is smaller than the size of the file, it is complemented by other dominated solutions taken from the population. The second situation occurs when the number of non-dominated solutions exceeds the size of the file. In this case, a truncation operator is applied. The solution that has the least distance from other solutions is removed from the set.

The selection procedure is made according to the fitness assigned to every solution using the strength value. The strength is defined as the number of solutions, in the archive A and the population P, that a particular solution j dominates. Thus, the fitness for a solution i is generated by the summation of the strength value of each solution, in the archive A and the population P, that dominates the solution i. Equation 2.11 shows how the fitness is calculated.

$$F(i) = \sum_{j \in P_t + A_t, j \succeq i} |\{k : k \in P_t + A_t \wedge j \succeq k\}| \tag{2.11}$$

If $F(i) = 0$ then this is a non-dominated solution. On the contrary, if $F(i)$ has a high value then this solution is dominated by a considerable number of other solutions. When two or more solutions have the same fitness values, the density information

D [321] is included in the selection process to refine the classification mechanism. Hence, solutions with a low density value are preferred to solutions with a high density value. Thus, the fitness is recalculated via the incorporation of the density information D as is shown in Equation 2.12.

$$F'(i) \quad = \quad F(i) + D(i) \tag{2.12}$$

SPEA2 has been applied successfully to solve some versions for multi-objective problems such as the knapsack problem and scheduling problem [321]. Also, it has been used to solve some applications in genetic programming and data mining [69]. However, the size N of the external file A and the truncation method to preserve diversity and maintain the size of the file A could affect the performance of the algorithm. This can occur when the number of non-dominated solutions found by the algorithm significantly exceed the pre-set size N and the number of objectives increases [98].

Algorithm 6 presents the general SPEA2 pseudocode.

Algorithm 6 Pseudocode of the SPEA2.

```
1: Generate the initial random population P and an empty external archive A
2: repeat
3:    Evaluate each individual from both the population P and the external archive
      A according to its strength
4:    Find all the non-dominated individuals from the population P and the external
      archive A
5:    if there are too many non-dominated solutions for the archive A then
6:       Use the truncation operator to remove individuals from A
7:    end if
8:    if there are not enough non-dominated individuals in the archive A then
9:       The best dominated individuals are selected to fill the archive A until it is
         full
10:   end if
11:   Apply the selection, crossover and mutation operators to the archive A to gen-
      erate the new population P
12: until the stop criteria is satisfied
13: return the non-dominated individuals from the archive A
```

2.5.2 Multi-Objective Evolutionary Optimisation Performance Measures

In multi-objective optimisation there has been two forms to compare approaches that are solving a particular problem [65]. The first of them has been used since the imple-

mentation of the first approaches to solve this sort of problem. The technique simply consists of comparing the plot for each approximated Pareto-front set and observing which of them is offering a better result. Although this mechanism is still being used to date, this is not useful when the visual comparison between two approximated Pareto-front sets are not so clear. Nowadays, new emerging approaches are more competitive and which are attempting more complex problems. As a consequence, the results comparison process is not as obvious as before. This is because not all the solutions belonging to one approximated Pareto-front set are better than the solutions in the other. Figure 2.10 illustrates an instance for both cases.

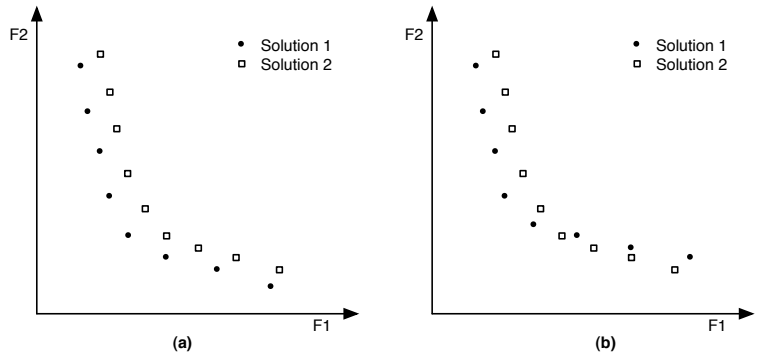


Figure 2.10: The Pareto-front set for a bi-objective problem where both objectives must be minimised. (a) Solution 1 is visibly better than Solution 2. (b) By just looking to the plot, it cannot be said which solution is better.

For this second instance, it is necessary to use a formal mechanism to determinate which solution is the best. Unfortunately, in this case there is no single criterion to establish whether an approximation set toward the true Pareto-front is better than another. Therefore, it is necessary to consider several criteria to carry out a good performance measure. Some of the criteria that could be considered are: the number of obtained solutions, the convergence or distance toward the optimum Pareto set, the diversity or distribution in the Pareto set, the relative coverage of the results over the objective space and the superiority comparison of a result upon the other. Thus, in order to implement a comparison between results that have been obtained by different approaches, it is necessary to define quality metrics. These metrics try to reflect, in a value, some criteria aforementioned in order to establish which approach performs better than others through obtaining higher quality results.

Nowadays, a series of metrics based on one or more criteria have been proposed which evaluate the quality of an approximation Pareto-front set. However, none of

these proposals is able to reflect all these criteria in a single value. For this reason, it is currently necessary to apply a number of quality metrics to have an objective assessment of the obtained results by different approaches.

Table 2.12 shows some of the metrics commonly used today, either by popularity or usefulness.

Criterion	Method Name	Autor(s)	Year	Description
Convergence	Error ratio (ER) [290, 293]	Van Veldhuizen	1999	The number of solutions in the non-dominated set A that are not members of the true Pareto-front set.
	Generational distance (GD) [291, 293]	Van Veldhuizen	1999	How far, on average, the non-dominated set A is from the true Pareto-front set.
Diversity	Spacing (SP) [258]	Schott	1995	Represent the spread of the solutions in the non-dominated set A.
Coverage	S-Metric [323, 318]	Zitzler	1999	How much of the objective space is dominated or covered by a non-dominated set A.
	C-Metric [318, 319]	Zitzler	1999	Represent the relative coverage comparison between two non-dominated set A and B.
	D-Metric [318, 319]	Zitzler	1999	Represent the relative coverage comparison between two non-dominated set A and B.
Superiority comparison	R1-Metric [139]	Hansen and Jaszkiewicz	2001	Represent the probability that a non-dominated set A is better than a set B over a set of utility functions.
	R2-Metric [140]	Hansen and Jaszkiewicz	2001	Represent how much a non-dominated set A is better than a set B over the difference between the expected values of the utility function.
	R3-Metric [140]	Hansen and Jaszkiewicz	2001	Represent the R2-Metris as a ratio.

Table 2.12: List of some methods used to measure the performance of multi-objective approaches.

From this metric list, five of them are chosen as performance metrics that could be applied to the multi-objective extremal optimisation approach proposed in this thesis. This selection is based on the implementation simpleness and the usefulness in the evaluation about the relevant features that are wanted to highlight in the proposed approach. Next, a brief description of these will be given.

2.5.2.1 Generational distance (GD)

The generational distance metric [291, 293] was proposed by Van Veldhuizen and Lamont. The aim of this metric is to calculate the approximate average distance between

the true Pareto-front set and the approximated Pareto-front set.

Equations 2.13 and 2.14 present the definition of this metric.

$$GD = \frac{\sqrt{\sum_{i=1}^{n} d_i^2}}{n} \tag{2.13}$$

$$d_i = \min_{j=1}^{m} \sqrt{\sum_{k=1}^{o} (f_j^i - f_j^k)^2} \tag{2.14}$$

where:

- n is the number of non-dominated solutions in the attained $\mathcal{PF}$ set,
- m is the number of non-dominated solutions in the true $\mathcal{PF}$ set,
- o is the number of objectives or dimensions,
- d_i is the Euclidean distance, in the objective space, between the solution i in the attained $\mathcal{PF}$ set and the nearest solution of the true $\mathcal{PF}$ set,
- f_j is the solution evaluate in the objective or dimension j.

When GD is zero it means that all solutions in the approximated Pareto-front set are in the true Pareto-front set. Any other value in GD reflects how far the approximated Pareto-front set is from the true Pareto-front set.

2.5.2.2 Spacing (SP)

The spacing metric [258] was proposed by Schott and measures how evenly the non-dominated solutions of the approximated Pareto-front set are distributed in the objective space. Equations 2.15, 2.16 and 2.17 present the definition of this metric.

$$SP = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (d_i - \overline{d})^2} \tag{2.15}$$

$$d_i = \min_{j=1 \wedge j \neq i}^{n} \sum_{k=1}^{o} |f_k^i - f_k^j| \tag{2.16}$$

$$\overline{d} = \frac{\sum_{l=1}^{n} d_i}{n} \tag{2.17}$$

where:

- n is the number of non-dominated solutions in the attained $\mathcal{PF}$ set,

o is the number of objectives or dimensions,

d_i is the distance, in the objective space, between the solution i and the nearest solution j in the attained $\mathcal{PF}$ set,

$\overline{d}$ is the average of all d_i,

f_k is the solution evaluate in the objective or dimension k.

If SP is zero then the solutions are evenly distributed along the approximated Pareto-front set. Also, it must be remembered that for the hypothetical case where the Pareto-front set is composed of only two solutions, this metric indicates that the result is evenly distributed. However, the number or density of solutions in the Pareto-front set is not representative.

2.5.2.3 S-metric

The S-metric [323, 318] measure was proposed by Zitzler and this indicates how much of the objective space is dominated by a given non-dominated Pareto-front set. This metric is useful to measure the coverage of a solution in an independent way.

Thus, for the approximated Pareto-front set obtained for a particular problem, the S-metric gives the area circumscribed by the union of the regions associated to each solution in the Pareto-front set. These regions are made up by the intersection of the consecutive hyperplanes over every solution along with the axes. Each objective is represented by a different axis. For instance, in a bi-objective problem, each solution generates a rectangle represented by the points $(0, 0)$ and $(f_1(s_i), f_2(s_i))$, where $f_1(s_i)$ and $f_2(s_i)$ is a non-dominates solution. Figure 2.11 shows the S-metric applied to a bi-objective problem for a minimisation problem.

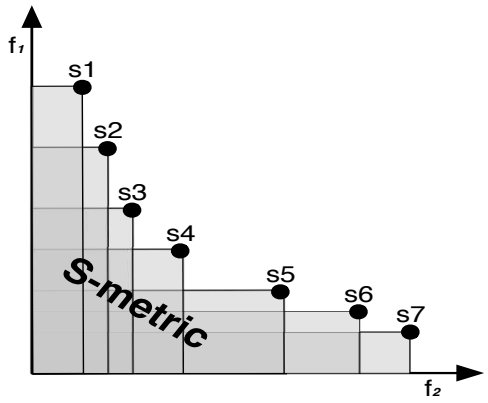


Figure 2.11: S-metric for a bi-objective problem.

2.5.2.4 C-metric

The C-metric [318, 319] was proposed by Zitzler and Thiele with the purpose of comparing two approximated Pareto-front sets (A and B) obtained by different approaches. This comparison is performed through the calculation of the fraction in which each set covers or dominates the other. As a result, two values are obtained. The first value is the percentage of solutions in B that are equal to or dominated by the solutions in A. The second value is the inverse case, the percentage of solutions in A that are equal to or dominated by the solutions in B.

A formal definition for this metric is shown in the Equation 2.18. The function C maps the ordered pair (A, B) into the interval $[0, 1]$.

$$C(A,B) = \frac{|\{b \in B; \exists a \in A : a \succeq b\}|}{|B|} \tag{2.18}$$

The nomenclature used is $C(A, B)$ and the interpretation is as follows. The value $C(A, B) = 1$ means that all solutions in B are dominated by A. The value $C(A, B) = 0$ represents the situation where none of the solutions in B are dominated by A. Both orderings have to be considered since $C(A, B)$ is not necessarily equal to $1 - C(B, A)$.

This metric just says if A dominates B but it says nothing about neither how good it is nor how even the distribution is. For example, the metric can give a better value for an approximated Pareto-front set A with a single solution, which is closer to the true Pareto-front set than another set B with various well distributed solutions but further from the true Pareto-front set. Another exceptional possible case is when the surface covered by the set A is equal to the surface covered by the set B however the set B, is closer to the true Pareto-front set than the set A.

2.5.2.5 D-metric

Originally, C-metric was developed to complement the aforementioned S-metric. However, to address some of the problems with C-metrics, Zitzler and Thiele proposed an improved indicator based on the coverage difference between two approximated Pareto-front sets A and B. The new D-metric [318, 319] is defined on the basis of S-metric and eliminates some present shortcomings in C-metric. The nomenclature used is $D(A, B)$ and this describes the size of the objective space that is dominated by A and is not dominated by B.

Equation 2.19 presents the definition for D-metric.

$$D(A,B) \quad = \quad S(A+B) - S(B) \tag{2.19}$$

Figure 2.12 shows an example for a bi-objective maximisation problem. For this problem $D(A,B) = \alpha$ represents the area covered only by the set A, $D(B,A) = \beta$ represents the area covered only by the set A and γ represents the area covered simultaneously by both set A and B.

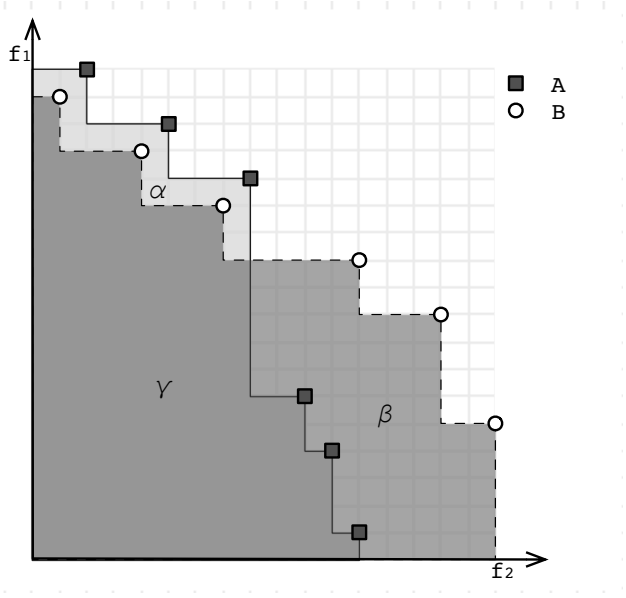


Figure 2.12: D-metric for a bi-objective maximisation problem.

In the case of using C-metric, $C(A,B) = C(B,A) = 0.5$ or 50%. That means there is no difference or supremacy in the coverage or dominance of one set over the other. However, with D-metric the quality gap between both sets can be reflected, so for the maximisation problem in Figure 2.12 $D(B,A) > D(A,B)$ which means that set B outperforms set A.

2.5.3 Multi-Objective for Extremal Optimisation

Until now, most of the metrics have limitations that do not allow to affirm conclusively whether one approach is better than another. For this reason, the most general recommendation observed in the literature is to use a set of metrics in such a way that these can interpret as many criteria as possible.

The selection of the five metrics described before is mainly based on the metrics used by the benchmark test applied in this thesis to observe the behaviour of the multi-objective extremal optimisation approach (S-metric, C-metric and D-metric). Also, two metrics are added (Generational distance and Spacing) to reinforce the study of the approach. These two metrics will be applied as a mechanism to clarify the goodness of one approach over another when necessary.

2.6 Summary

Nature-inspired meta-heuristics are an effective and efficient tool to solve combinatorial optimisation problems that have large and complex search spaces. One of the most paradigmatic bio-inspired meta-heuristics are genetic algorithms. These algorithms, that are based on the theory of evolution of species by natural selection, have been extensively applied into theoretical and real world problems, becoming a reference method to validate new nature-inspired approaches.

With a more comprehensive point of view, memetic algorithms are an extended meta-heuristic from evolutionary algorithms based on the cultural evolution. This meta-heuristic subscribes to the hybridisation of different algorithmic approaches to solve a particular problem. The successful results achieved by the synergic use of diverse search techniques, as an integrated algorithmic approach, are encouraging the extension of other methods, different to evolutionary algorithms, toward the memetic algorithms scheme.

From the swarm intelligence branch in nature-inspired meta-heuristics, ant colony optimisation is an interesting multi-agent method highly suitable for problems where it is necessary to connect and/or route two or more elements through a path in an optimal way. As its name suggests, this scheme is based on the ants' behaviour when they transport food from the source to the nest. The implementation of the pheromone concept in this meta-heuristic is a significant and clever idea that may be incorporated into other methods as a memetic operator.

Extremal optimisation is a relatively recent, novel and emerging nature-inspired meta-heuristic whose search method is especially suitable to solve combinatorial optimisation problems. This method has the peculiarity that it is based on both biological and physical inspiration. Many of the problems solved using extremal optimisation can be categorised as unconstrained problems and a relatively small amount of research work has been done on constrained combinatorial optimisation problems. Additionally, most of the research in extremal optimisation has been applied to solve single-

objective problems and only a relatively small number of attempts have been done to extend extremal optimisation toward multi-objective problems. The implementation of a hybrid extremal optimisation framework with a constraint handling technique to solve constrained combinatorial single-objective and multi-objective problems is a relevant contribution to the extremal optimisation method and a challenging proposal in the field of nature-inspired meta-heuristics.

Various techniques have been developed in evolutionary algorithms to handle the presented constraints in some optimisation problems. These techniques are classified in five different groups. Thus, the *superiority of feasible points* mechanism from the *separation of constraints and objectives* group seems to be a natural way to be applied as a constraint handling technique to extremal optimisation. The superiority of feasible points uses a ranked order for the fitness evaluation, where a better value is assigned to feasible solutions and a worse value to infeasible solutions. This scheme fits naturally with the rank mechanism in extremal optimisation which sorts the fitnesses from the poorest to the best.

The evolutionary multi-objective optimisation area has been developing a number of competitive methods to solve multi-objective problems in the past two decades. From these, the most representative and well studied methods are SPEA2 and NSGA-II which are de-facto reference methods to compare new approaches. It is necessary to perform the comparison between approaches to carry out performance measures through the evaluation of some metrics. These metrics are based on criteria such as convergence, diversity or coverage through which it aims to define which method produces the best results. Finally, the validation for the proposed multi-objective hybrid extremal optimisation framework is able to be done through the use of the mechanisms and tools presented for multi-objective optimisation.

Chapter 3

A Single-Objective Hybrid Extremal Optimisation Framework with Constraint Handling

3.1 Introduction

Extremal optimisation is a single-solution meta-heuristic based on a bio-inspired co-evolutionary model which in turn is based on the physics-inspired scheme of self-organised criticality [14]. Extremal optimisation has been principally applied to problems in physics and its implementation has been compared with other methods such as simulated annealing and genetic algorithms. Recent research into extremal optimisation has expanded its application toward new combinatorial problems such as assignment type problems and dynamic problems. Having regard to all sorts of problems solved by extremal optimisation to date, it can be noted that most of them may be categorised as being unconstrained. It is evident that there exists a necessity for incorporating, in a methodical manner, a constraint handling mechanism into extremal optimisation to solve constrained combinatorial optimisation problems. This should be done by taking into consideration the exploitation of the natural algorithmic mechanics of extremal optimisation.

Recall that the principle behind extremal optimisation is to eliminate one of the weakest or least adapted species and replace it by a random one at each iteration. This stochastic characteristic of extremal optimisation was studied by Hendtlass and

Randall [143]; and Randall, Hendtlass and Lewis [244] on assignment type problems. One of the main conclusions from these works was that, because of the feature of extremal optimisation to escape from local optima, its capacity to converge toward the optimal solution is more restricted. To overcome this lack of convergence, the application of a local search mechanism was tested. Results proved it to be a good option to improve the search for the optimum value. Based on this evidence, this thesis will incorporate a *secondary search mechanism* to complement extremal optimisation and improve its ability to converge. This enhancement has also been a common tactic used by other meta-heuristics.

Given the above mentioned arguments, the design of a *hybrid extremal optimisation* (HEO) framework with constraint handling is presented herein. It is designed to find solutions through the collaborative use of extremal optimisation with a secondary search and constraint handling mechanisms. For this research, the chosen secondary search mechanism is represented by a local search (LS) scheme. The stochastic nature of the extremal optimisation component carries out a coarse-grain search on the landscape, allowing the solution the possibility of escaping from stagnant positions to potentially move between feasible and infeasible spaces. A *differentiated fitness evaluation* is performed depending on whether the current solution is feasible or infeasible. The secondary search mechanism helps to develop a fine-grain movement in the neighbourhood area to obtain a better approximation of the optimal solution. Thus, the motivation to use extremal optimisation is influenced by the promising results obtained in previous research, which emphasises its simplicity of implementation due to its single parameter and selection procedure.

Aware of the large number of types of combinatorial optimisation problems that can be found, the development of this framework is initially aimed at those problems that have capacity constraints. Problems with this type of constraint generally come with the necessary information to perform a *component fitness evaluation*. This decision also is biased by the analysis performed on extremal optimisation by Randall [243], Hendtlass and Randall [143] and Randall, Hendtlass and Lewis [244] that outline the potential of this method to solve such problems. Figure 3.1 shows representative capacitated combinatorial optimisation problems that could be solved with the hybrid extremal optimisation framework.

From these problems, the multi-dimensional knapsack, the bin packing and the generalised assignment problems are selected to be completely implemented and analysed. This selection is based on the representativeness of these problems, as many real-world problems are derivatives of these. Also, the knowledge of these problems

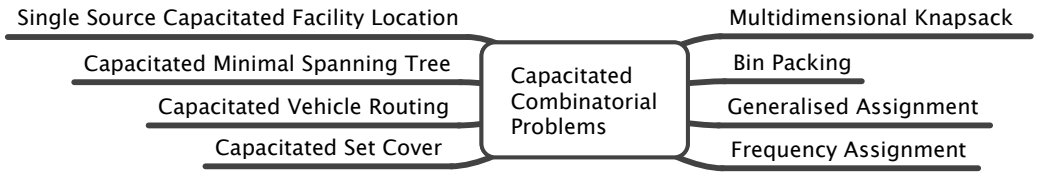


Figure 3.1: A representative sample of capacitated combinatorial problems.

is extensive, as they are three of the most well-studied combinatorial problems from the above list [72, 80, 95, 159, 240, 260, 262]. Later on, in this chapter, the HEO implementation of the remaining five problems will be discussed. Additionally, the way of representing the solution as an array for each problem, by either a vector of binary numbers or a vector of positive integer numbers, presents the possibility to observe both the factors that are present in the modelling of the problem and the effectiveness of finding good solutions. To trial the three main problems, benchmark instances and data for each problem from the OR-Library [24] are selected. Also, from OR-Library, benchmark instances and data for the frequency assignment, the single source capacitated facility location and the capacitated minimal spanning tree, can be found. In VRP Web [106] there exists a complete set of benchmark instances and data for the capacitated vehicle routing problem. However, for the capacitated set covering problem it was not possible to find a public availability of test instances.

This chapter pursues the following two objectives:

1. To incorporate a constraint-handling mechanism that allows extremal optimisation to deal with infeasible solutions.

2. To provide a hybrid extremal optimisation framework to solve single-objective constrained combinatorial optimisation problems.

The rest of this chapter is organised as follows. Section 3.2 explains the hybrid extremal optimisation framework to solve constrained combinatorial optimisation problems. Section 3.3 shows how the hybrid extremal optimisation is applied to the constrained combinatorial optimisation problems presented in Figure 3.1. Section 3.4 presents a summary of the computational experiments developed with an analysis of the three selected problems. Section 3.5 summarises the chapter and discusses the future work arising from this study. Finally, Section 3.6 states the contributions that arise from this chapter.

3.2 Hybrid Extremal Optimisation with a Constraint Handling

This section describes the hybrid extremal optimisation framework. First, the proposed constraint-handling scheme for extremal optimisation is defined. Then, the secondary search mechanism, as a complement to extremal optimisation, is explained. Finally, the integrated framework to deal with constrained combinatorial optimisation problems is described.

3.2.1 Constraint Handling for Extremal Optimisation

Extremal optimisation was proposed originally to give more efficient solutions to the spin glass problem in physics, which is associated with random magnetic fields [312]. To solve this problem, it was modelled using graphs as a combinatorial max-cut problem, which has only feasible solutions. After this, extremal optimisation has been mostly used for solving combinatorial problems where the representation of the solution does not generate infeasible solutions (as outlined in Chapter 2). This trend is due to the selection mechanism not being originally designed to handle infeasible solutions.

However, for many constrained combinatorial problems, it is not possible to represent the solution, in such a way, that only feasible solutions are generated. In this case, the search mechanisms applied to such problems must be aware of how the search space is traversed. This is because the search space, in constrained problems, is divided into two regions. One region contains the feasible solutions and the other contains infeasible solutions (see Figure 2.7 on page 37). Two possible ways to address this scenario are: a) focus the search only in feasible regions; or b) allow the search to traverse feasible and infeasible regions. In the former case, infeasible solutions are eliminated or transformed into feasible solutions. In the latter case, the solution is able to move throughout the search space passing through feasible and infeasible regions looking for the optimal solution. This case is the one that fits more instinctively with extremal optimisation because of its stochastic nature. Thus, taking into account the second point of view, extremal optimisation will be extended to deal with the restrictions that constrained combinatorial optimisation problems have.

Extremal optimisation and evolutionary algorithms have some similarities that are relevant for this research. Both methods evolve across generations, have a selection operator based on a fitness evaluation, and the replacement mechanism in extremal optimisation can be seen as a mutation operator. These three similarities enable ex-

tremal optimisation to incorporate and use many of the constraint handling techniques used by evolutionary algorithms (described in Section 2.4). Therefore, it is necessary to analyse which of these techniques are most suited to be incorporated into extremal optimisation.

Considering the five groups mentioned in the constraint handling techniques in Section 2.4, three of them are most commonly used by search algorithms. They are the *penalty functions*, *repair algorithms*, and *separation of constraints and objectives*. However, in the latter can be found the *superiority of feasible point* technique proposed by Powell and Skolnick [239] and also by Deb [96]. This technique ranks solutions, giving better fitness values to feasible solutions and worse fitness values to infeasible solutions. Through this ranked order fitness evaluation, an infeasible solution will never obtain a better evaluation than a feasible solution, as may be the case with penalty functions. This rank selection method for the solutions for genetic algorithms has common features with the rank selection method for the components of the solution in extremal optimisation. A relevant feature of this technique is its differentiated evaluation mechanism. The fitness value for a feasible solution is calculated using the objective function and for an infeasible solution it is calculated using the constraints. This separated evaluation makes it possible for feasible solutions to move toward the optimal point and for infeasible solutions to move into a feasible region. In the case of a single-solution search mechanism such as extremal optimisation, this evaluation scheme enables the solution to move along feasible and infeasible regions searching for the optimal value. Thus, this differentiated fitness evaluation scheme is completely compatible and fits naturally within extremal optimisation's algorithmic mechanics. So when the solution is in an infeasible region, the differentiated fitness evaluation will allow it to change the direction of its search (in a natural and smooth way) toward a feasible region. Figure 3.2 illustrates these movements.

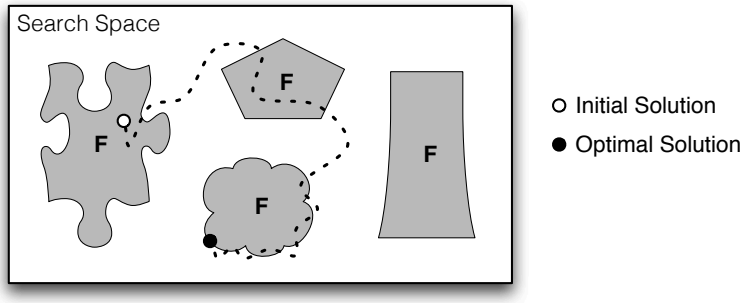


Figure 3.2: Solution movements through the search space crossing feasible and infeasible regions. The shaded areas represent feasible space while the white areas are infeasible.

The general form to represent the fitness evaluation for this technique is shown in Equation 3.1.

$$\lambda(x) = \begin{cases} f(x) & \text{if the solution is feasible} \\ p \pm \sum_{i=1}^{n} c_i(x) & \text{if the solution is infeasible} \end{cases} \tag{3.1}$$

where:

$\lambda(x)$	is the fitness value of solution x,
$f(x)$	is the objective function evaluated for the feasible solution x,
$c_i(x)$	is the i^{th} constraint applied to the infeasible solution x,
p	is a parameter defined for a particular problem that separates the evaluation of feasible and infeasible solutions.

A problem with this technique is a lack of diversity in the offspring population due to the superiority of feasible solutions over infeasible solutions. However, the inherent drawback in this constraint handling technique is that it is not valid for extremal optimisation as it is a single-solution method. Thus, diversity of the offspring is not a concern since extremal optimisation has no crossover operator and does not converge.

In terms of the other two techniques, penalty functions and the repair algorithms, their implementations in extremal optimisation are not so direct. A penalty function approach puts the objective and the constraints together in a new objective function in which it is necessary to define the correct penalty factor to obtain an appropriate fitness evaluation. There are many techniques to build penalty functions that derive more suitable penalties. The more accurate techniques can find better solutions but with greater complexity in their implementation than is usually reflected in the use of several tuning parameters. Repair algorithms are appropriate when the conversion from an infeasible solution to a feasible one does not require too much effort. On the contrary, the cost of reparation could increase enormously, degrading the whole performance of the algorithm. Also, most repair techniques are very problem dependent [64].

Taking into account what has been explained above, the superiority of feasible point technique in the *separation of constraints and objectives* group seems to be a reasonable alternative to be applied as a general and simple constraint handling mechanism to extremal optimisation. This technique is able to maintain the natural characteristic of extremal optimisation to move across feasible and infeasible regions looking for the best solution. But most importantly, this technique allows extremal optimisation to properly define the way to assess the fitness value for each component

of the solution when it is either feasible or infeasible. Recall that extremal optimisation generates new solutions by changing the values of poor components; that is, it seeks which of the components gives a poor contribution or degrades the solution the most. Hence, when the current solution is feasible, the feasible fitness function must be focused on locating which component of the solution adversely affects its quality the most with the objective to increase its optimality, ideally reaching the optimal solution. However, when the current solution is infeasible, the infeasible fitness function must be focused on locating which part of the solution is more responsible for the violation of one or more constraints. This is aligned to the objective of reducing the degree of infeasibility and so ideally converting it to a feasible solution.

Based on the superiority of feasible point technique, a constraint handling extension for extremal optimisation is proposed herein, which hereafter is referred to as the "differentiated fitness evaluation scheme". In order to achieve this, it is necessary to adapt the original superiority of feasible point technique to assess each component of the solution and not the entire solution.

Defining the component of the solution and its contribution, from the point of view of optimality or infeasibility, is not an easy task. First, it is necessary to clarify which elements from the solution representation will be considered as the evaluation components. Subsequently, these components are analysed to calculate their contribution, positive or negative, through the fitness function.

For example, in the case of an unconstrained problem with a 0-1 binary representation, only the component contribution with respect to the objective function is required. A simple way to carry out the component evaluation is switching the value for each component in the solution and evaluating it using the objective function. With this result, components can be ranked by taking into account the improvement in the objective function (depending on whether it is a maximisation or minimisation problem). From the ranked list, the extremal optimisation selection mechanism is applied to choose the component to be eliminated and changed. However, the process described above has simply evaluated, via the objective function, a number of neighbouring solutions generated after changing the value of each component from the current solution instead of measuring the contribution of each component. This procedure could produce an unnecessary increase in runtime. Additionally, in a constrained problem, the satisfaction or violation of the restrictions associated with the problem also must be considered in the evaluation process. To this should be added that many of the problems may be evaluated in a different way according to the information or data available and the particular characteristics of each.

When the component of the solution has an associated coefficient of cost, profit or any other that must be minimised or maximised, the *shadow price concept* [83, 172] used in linear programming can be used to measure the contribution of each component in the solution. The shadow price can be understood as the value that it is willing to pay for an additional unit of a particular limited resource. In other words, the shadow price represents the marginal utility or the marginal cost in the objective function that can be attained when a constraint is increased or decreased in one unit for a maximisation or minimisation problem.

In extremal optimisation, the shadow price can be defined as the potential contribution (positive or negative) that can be provided to the objective function when a component of the solution undergoes a change. This change in the component can be represented by a modification in either its value or its position. To use the shadow price idea as the fitness function for constrained combinatorial problems in extremal optimisation, it is necessary to measure the component contribution with respect to the feasibility and the optimality of the solution.

When the component of the solution does not have an associated coefficient, its contribution to the feasibility or infeasibility of the solution or to the minimisation or maximisation of the objective function could be assessed by any coefficient associated to the values or elements that can be assigned to each component.

Defining a general fitness function for the components of the solution instead of the entire solution could become a hard task because not all problems provide the necessary information to perform this particular evaluation. However, in an attempt to define a common formula to carry all this out, a general, simple and comparable description to assess the contribution among the components of the solution using the differentiated fitness evaluation scheme and the shadow price concept is described by Equation 3.2.

$$\lambda(x_i) = \begin{cases} -(S_i \pm w_i) & \text{if the solution is feasible} \\ V_i \pm w_i & \text{if the solution is infeasible} \end{cases} \tag{3.2}$$

where:

- S_i represents the constraint satisfaction level for component i,
- V_i represents the constraint violation level for component i,
- w_i represents the coefficient associated with component i (profit or cost) in the objective function, when this value is available.

The fitness evaluation is composed of two parts. The first is an integer value that represents the potential level of either satisfaction S_i or violation V_i with respect to the constraints of the problem when the i^{th} component of the solution changes its current value.

For a feasible solution, the satisfiability could be assessed as the number of remaining satisfied constraints and/or the availability of the constrained resources. For example, in a capacitated problem with five constraints, if all constraints remain satisfied when the i^{th} component change its current value to any other value within the legal scope of alternatives, then S_i is set to 5. If one the constraints is not satisfied then S_i is 4 and so on to reach the value 0 when none of the constraints are satisfied. Now, if the problem has only one constraint with a capacity of 100 units, then S_i is the number of available units that the constrained resource can still contain (e.g., S_i=20 means that the resource has room for 20 units) when the i^{th} component change its current value to any other value within the legal scope of alternatives. In this case S_i could have a value less than 0 when any change in the i^{th} component causes the constraint to be unsatisfied.

In the case of an infeasible solution, the violation could be measured as the number of violated constraints and/or the degree to which a restriction has been violated. Thus, following on from the above example of the capacitated problem, a new scenario can be evaluated when four of the five constraints are violated. In this case, if a violated constraint becomes satisfied when the i^{th} component change its current value to any other value within the legal scope of alternatives, then V_i is set to 3 and so on to reach the value 0 when none of the constraints are violated. On the other hand, when the problem has only one constraint, then V_i represents the number of units that overload the resource (e.g., V_i =50 means that the resource is overloaded by 50 units) when the i^{th} component changes its current value to any other value within the legal scope of alternatives. In this case V_i could have a value less than 0 when any change in the i^{th} component causes the constraint becomes feasible.

The second part, w_i, is a decimal value into the interval [0, 1] that represents the normalised profit or cost value associated with the i^{th} component. This second component has the aim of refining the evaluation when two or more components have an equal value in the first part.

Thus, when the solution is feasible, a component with a fitness $\lambda(x_i)$ that maintains a high level of satisfiability S_i and has a high profit (maximisation problem) or low cost (minimisation problem) w_i can be considered to be degrading the solution. This is because if the value of the i^{th} component changes, then there is a greater chance that

it provides a better contribution to the solution. On the other hand, when the solution is infeasible, a component with a fitness $\lambda(x_i)$ that achieves a low level of violation V_i and has a low profit (maximisation problem) or high cost (minimisation problem) w_i can be considered to be degrading the solution. This is because a modification on the value of the i^{th} component can provide a better contribution towards the feasibility of the solution.

The proposed constraint handling technique for extremal optimisation illustrated in Figure 3.3 could be complemented with the repair algorithm technique in the case that the solution takes too much time to return to a feasible space. Thus, a repair technique such as the partial feasibility restoration algorithm proposed by Randall [243] could be used to accelerate the convergence of the infeasible solution when the problem to be solved has difficult constraints to satisfy.

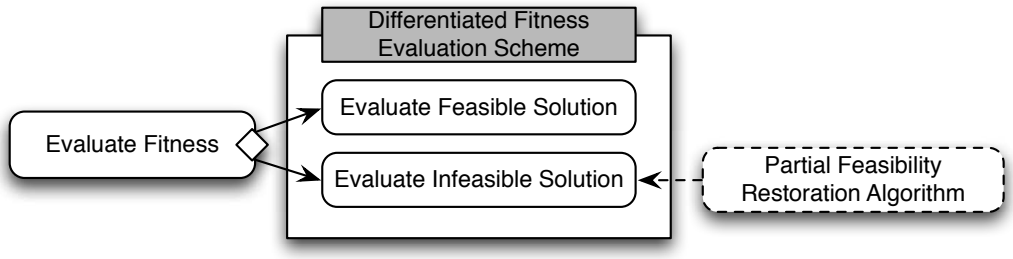


Figure 3.3: The constraint handling scheme for HEO.

3.2.2 A Secondary Search Mechanism for Extremal Optimisation

As stated previously, one of the main characteristics of extremal optimisation is its ability to move through feasible and infeasible regions looking for the optimal solution without being trapped in local optima. However, this characteristic also hinders its capacity to refine the search when a solution is near the optimal result. Therefore, it is necessary to complement the search process in extremal optimisation with a secondary mechanism that could be executed when a feasible solution is found. There, of course, exists previous research concerning extremal optimisation applied to combinatorial optimisation problems that has shown the advantages of complementing it with local search mechanisms [130, 131, 143, 244].

The stochastic nature of extremal optimisation is reflected in a coarse grain exploration over the search space. This exploration can be refined using an additional

search algorithm, as a secondary mechanism, that allows a fine grain inspection of the neighbourhood that is associated with the last feasible point found by extremal optimisation. This secondary search mechanism should ideally be one that does not significantly degrade the agile performance of extremal optimisation. Some mechanisms that emerge as alternatives which may be incorporated are: local search, tabu search, simulated annealing, and the (1+1) evolutionary strategy.

Thus, every time that extremal optimisation generates a new feasible solution, the secondary search mechanism could be activated to move the current solution closer toward the optimal solution. Figure 3.4 illustrates a schema of the secondary search mechanism proposed to complement the canonical extremal optimisation metaheuristic.

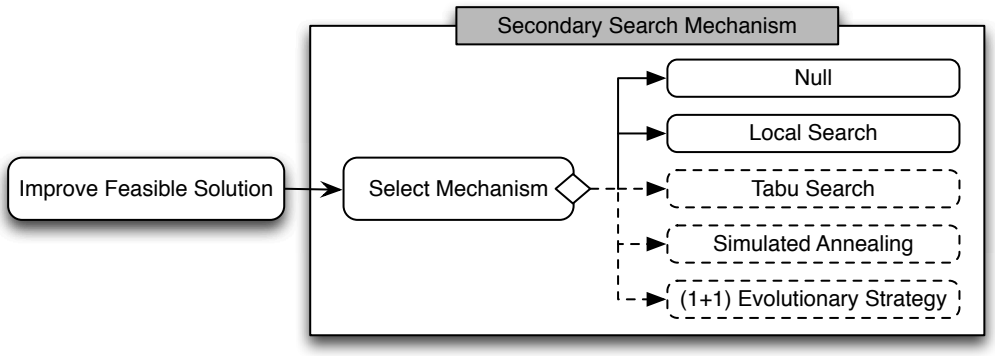


Figure 3.4: The secondary search scheme proposed to complement EO. The mechanisms with continuous lines will be developed in this thesis. The mechanisms with dotted lines will be subject to future research.

This scheme follows the memetic algorithms concept [217, 218]. The idea is to apply one of the available secondary search mechanisms from the collection of mechanisms present in the framework. However, it is not compulsory that the chosen technique must be applied throughout the entire evolutionary process. One criterion to decide if either a different secondary search mechanism is to be used or none at all (i.e. Null) could be applied based on the measure of the improvement achieved by the current technique in use. If there is no improvement observed in a given period of time, a change in the secondary search mechanism to a different one could be made.

Local search has been chosen for this research as the initial algorithm to be incorporated into the secondary search mechanism scheme. The reason for the selection of the local search technique is based on its feature to detect local optima efficiently, which the previously mentioned mechanisms cannot guarantee. Also, local search is a mech-

anism used by many evolutionary meta-heuristics that have reported good results in improving the quality of solutions that they are generating (e.g. [13, 32, 219, 220, 317]). The search is commonly implemented as an iterative process that starts with a feasible solution and then improves it by performing a sequence of small modifications. This process is repeated until the current solution cannot become better.

The exploration of new solutions in a combinatorial optimisation problem generally is performed through the change in at least one of the components of the solution. This change can be represented either by an alteration in a particular component or by a movement or swapping between two components. Local search techniques apply a series of these changes to the solution components looking for new configurations that will improve the current solution in a greedy way.

The extremal optimisation search can be seen as a sequence of alterations on the components that degrade the solution. For this reason, the local search algorithm will be aimed at developing the movement or swapping between the components of the solution. Thus, extremal optimisation and local search perform a different sort of change in the solution in order that both methods can work in a complementary way.

Taking, as a reference, the swapping idea from the tailor-made local search technique for the bin packing problem developed by Falkenauer [112], a novel general purpose local search is proposed. This approach performs the swapping based on a double traverse mechanism to explore the representation of the solution. First, the algorithm looks for the possibility of swapping one component linked to the first traverse by one component linked to the second traverse. Second, the algorithm looks for the possibility of swapping one component linked to the first traverse by two components linked to the second traverse. In both cases the swap is accepted provided that the result of this movement produces an improvement in the objective function without violating any of the problem's constraints. This procedure is repeated until it is not possible to generate any further improvement to the solution. Note that the way the double traverse is carried out on the solution, and the information swapped between the components depends on the problem to be solved and how the solution is represented. Also, not all swaps can be feasibly performed, This will depend on diverse aspects such as the particular interpretation given to the components, the type and range of values assigned to the components and the possible relationship between components with respect to the constraints of the problems.

The double traverse performed to implement the local search mechanism naturally fits with the vector structure used to represent the components of the solution in extremal optimisation. Also, the swapping movement carried out can be seen as a

more elaborate and extended replacement operator than the simple operator used in extremal optimisation. The double traverse local search is an $O(n^2)$ algorithm where n is the number of components of the solution.

Algorithm 7 shows a general *double traverse local search* pseudocode as a secondary search mechanism to complement the extremal optimisation meta-heuristic.

Algorithm 7 The general double traverse local search pseudocode for CCOPs.

```
1:  repeat
2:    {First Step}
3:    for each component i in the solution structure do
4:      for each component j different from i in the solution structure do
5:        if the swap between components i and j is possible AND the solution
          remains feasible AND as a result of this operation a better solution is
          generated then
6:          Implement the operation and update the variables involved in the action
7:        end if
8:      end for
9:    end for
10:   {Second Step}
11:   for each component i in the solution structure do
12:     for each components j and j' different from i in the solution structure do
13:       if the swap between component i and components j and j' is possible
          AND the solution remains feasible AND as a result of this operation a
          better solution is generated then
14:         Implement the operation and update the variables involved in the action
15:       end if
16:     end for
17:   end for
18: until there is no improvement in the solution
```

3.2.3 Hybrid Extremal Optimisation Framework

The hybrid framework is based on the extremal optimisation meta-heuristic that, in its canonical version, only works for unconstrained problems. In this chapter, an initial framework is proposed that enhances the canonical version by adding two new characteristics. These characteristics are a differentiated fitness evaluation scheme to extend extremal optimisation to solve constrained combinatorial problems and a secondary search mechanism to support the extremal optimisation mechanism through the refinement of feasible solutions with the objective of improving the convergence toward the optimal solution. This framework allows new features to be incorporated,

as they are needed to solve new types of problems. This is evidenced by the proceeding chapters. Figure 3.5 presents the novel general hybrid extremal optimisation framework and a more detailed description of the framework will be given below.

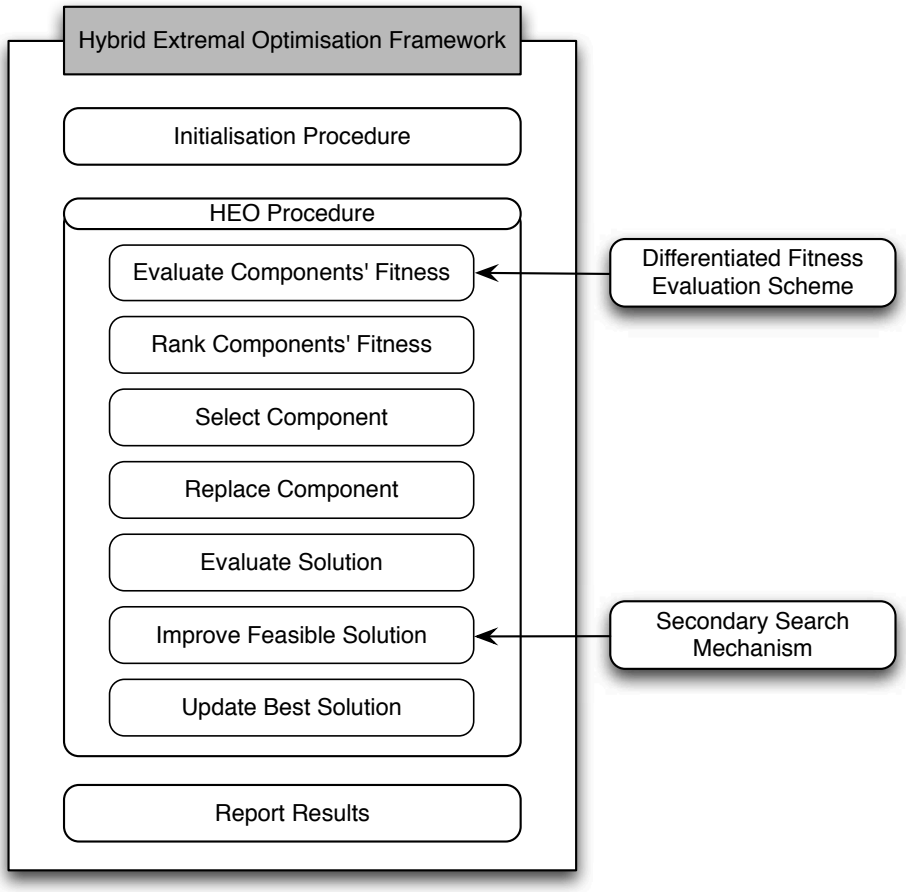


Figure 3.5: The Hybrid Extremal Optimisation (HEO) Framework to solve single-objective constrained combinatorial problems.

The hybrid extremal optimisation framework starts with the initialisation procedure. This initialises the variables and structures that will be required to begin the hybrid extremal optimisation process. First, the probability vector P, which depends on the parameter τ to generate the power law distribution according to Equation 2.5, is generated. Note that this probability vector will be used later for choosing the component value to be changed in the solution and to be replaced subsequently by a new value.

The framework requires that an initial solution S be generated. This initial solution could be created in many diverse ways; for instance, a random selection of the

components of the solution seems to be a common and simple choice. However, this is likely to result in an infeasible solution, particularly if the problem is moderately or highly constrained. Therefore, starting the process with a feasible solution may avoid a potential waste of runtime; however, the cost of generating a feasible solution must be considered. Also, some problems could require the generation of a tailored initial solution with the aim of improving the performance of the algorithm or by the requirement of a particular problem, which must start with a specific configuration for the initial solution. For example, a predefined collection of investments in the optimisation of a portfolio using a model based on the knapsack problem might be needed by a financial organisation.

If the generated initial solution S is feasible then this becomes the first best solution, S_{best}, so far; otherwise, S_{best} is initialised with an extreme value (high or low) depending on whether it is a minimisation or maximisation problem respectively. Figure 3.6 shows the scheme for the initial solution procedure proposed for the hybrid extremal optimisation framework.

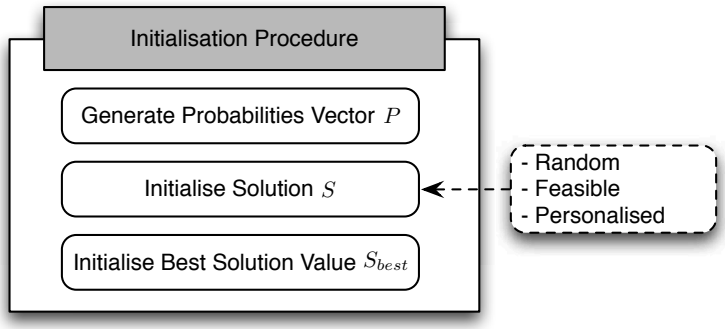


Figure 3.6: The initialisation procedure for HEO.

Once the initialisation process is completed, the iterative hybrid extremal optimisation process begins. Recall that when a constrained combinatorial optimisation problem is being solved using extremal optimisation, the solution can move between feasible and infeasible space (according to Randall [243]) depending on the particular features of the search space associated with to each problem. Thus, it is necessary to carry out an appropriate extremal optimisation selection process for both cases.

When the solution is feasible, the fitness calculation is focused on locating the part of the solution that adversely affects its quality the most. However, when the solution is infeasible, the fitness calculation is focused on locating which part of the solution is more responsible for the violation of one or more constraints. Section 3.2.1 discussed

how the fitness is calculated for constrained combinatorial optimisation problems using the proposed differentiated fitness evaluation scheme as the constraint handling technique.

After calculating the fitness for each component of the solution, they are ranked from worst to best. To perform this ranking, a sorting technique must be applied to list the components in order from lower to higher fitness values. Figure 3.7 presents four of the most efficient and effective sorting approaches that could be applied to perform this task. They are: quicksort [145], heapsort [303], mergesort [171] and introsort [226].

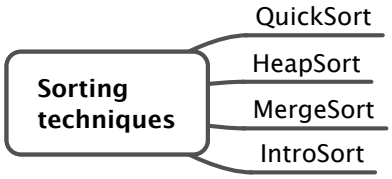


Figure 3.7: Sorting techniques for HEO.

Quicksort enjoys a large popularity due to its high efficiency, as it is an algorithm suitable for most of the average instances with an excellent average performance. Hence, the quicksort technique will be used in this research.

Once the components have been ranked, one of them is selected to replace its value for a different value based on the probability of its rank in P, using some selection technique. The selection technique can be chosen from an existing set available from the literature. Among the most used are roulette wheel selection [148], tournament selection [127], ranking selection [18], and stochastic universal sampling [18]. Figure 3.8 shows the selection techniques that could be applied in the hybrid extremal optimisation framework.

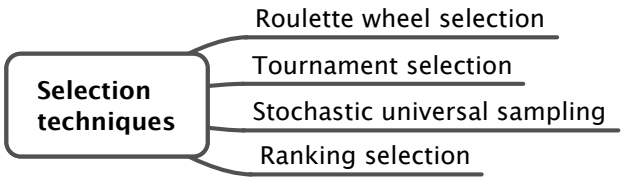


Figure 3.8: Selection techniques for HEO.

Considering that extremal optimisation works with a single-solution scheme, roulette

wheel selection and ranking selection are the most suitable techniques to be applied. However, in the previous step of the extremal optimisation algorithm, a ranking of the components based on their fitness has already been carried out. Therefore, the roulette wheel selection is the most suitable option to be used with extremal optimisation.

To replace the data of the selected component, in the canonical extremal optimisation, a new random value, within the legal scope of alternatives, is chosen. When this new value is assigned to the selected component, a new solution configuration is generated. However, this is not the only procedure that could be applied as the component replacement mechanism to generate new solution configurations. Instead of generating a random value, this could be chosen by applying the selection rules of extremal optimisation. Thus, from the set of legal values that can replace the data of the selected component, one of them is chosen according to how much the component degrades the solution with the assignment of this new value. This may lead to a more direct path toward the optimal solution because the component replacement mechanism will be biased to choose values that improve the quality of the solution.

Next, the new solution is accepted unconditionally and both its objective function evaluation and its feasibility state are updated. In the case that a feasible solution is obtained, the secondary search mechanism procedure (refer to Section 3.2.2) is called to see whether it is possible to find a better solution in the neighbourhood of the current solution. Once the secondary search is completed, the current solution is analysed to determine if a better solution has been reached. If so, the new found solution is saved as the best solution discovered so far.

The HEO method is repeated until a termination condition is reached. The terminate criteria to finish the extremal optimisation evolution can be chosen from the following:

- *Generation number.* A pre-set number for the maximum number of generations is defined. If the process reaches this maximum number then the evolution stops.

- *Evolution time.* A pre-set runtime for the maximum duration of evolution is defined. If the computational program reaches this maximum time then the evolution stops.

- *Best solution time.* A pre-set period of runtime is defined. If in that period the best solution found so far has not changed, the process is terminated.

- *Personalised.* Evolution stops when a personalised criterion is achieved. This makes sense when a special requirement is mandated during the execution of

the process. For example, finish the process when a memory limit is exceeded or when the solution reaches a predetermined result.

Finally, the best-found solution by the hybrid extremal optimisation process and its evaluation are returned in a file for later analysis. Algorithm 8 shows the pseudocode of the hybrid extremal optimisation framework for single-objective constrained combinatorial optimisation problems.

Algorithm 8 The HEO framework for single-objective CCOPs.

{**Initialisation Procedure**}
Generate the probability vector P
Initialise solution S
if S feasible **then** $S_{best} \leftarrow S$
{**HEO Procedure**}
repeat
  if the current solution is feasible **then**
    Evaluate the fitness $\lambda(x_i)$ for the components of the feasible solution
  else
    Evaluate the fitness $\lambda(x_i)$ for the components of the infeasible solution
  end if
  Rank the components according to their individual fitnesses $\lambda(x_i)$ from the worst to the best
  Select a component based on the probability of its rank P using any selection technique
  Obtain a S_{new} in the neighbourhood of S when the selected component value is replaced by a different one
  Evaluate the new solution S_{new}
  if S_{new} is feasible **then**
    {**Secondary Search Mechanism Procedure**}
    S_{new} = Apply the secondary search mechanism to S_{new}
    if the evaluation of S_{new} is better than the evaluation of S_{best} **then**
      $S_{best} \leftarrow S_{new}$
    end if
  end if
until the termination condition is satisfied
return S_{best} and the evaluation of S_{best}

3.3 HEO Applied to CCOPs

This section explains how the hybrid extremal optimisation framework is applied to the $\mathcal{NP}$-hard constrained combinatorial optimisation problems shown in Figure 3.1.

First, a concise description for the multi-dimensional knapsack, the bin packing and the generalised assignment problems, which will be used to test the new extremal optimisation approach, is developed. These problems were chosen because they are amongst the most well-studied in the literature. As identified in Chapter 2, they form the basis of many different applications.

For the remaining five problems, the relevant aspects about how HEO can be applied to them are also described.

3.3.1 Multi-dimensional Knapsack Problem (MKP)

The knapsack problem is a classic benchmark problem with which many real-world problems can be represented. For this reason, it is one of the most analysed problems in combinatorial optimisation [285]. There exist many variants of this problem according to the particular case being solved. However, the multi-dimensional version has been chosen because of the high degree of constrainedness present in its multi-dimensional form.

The multi-dimensional knapsack problem is an $\mathcal{NP}$-hard combinatorial optimisation problem and is a generalisation of the knapsack problem, which has only one constraint. The aim of the multi-dimensional knapsack problem is to find a combination of n items that can maximise profit without exceeding the amount of resources allowed by the m constraints. Generally, these restrictions symbolise different limitations, for instance, in weight, length, volume, and cost. Formally, it can be represented as:

$$\begin{aligned} \text{Maximise} \quad & \sum_{i=1}^{n} p_i x_i \\ \text{Subject to} \quad & \sum_{i=1}^{n} r_{ij} x_i \leq b_j \quad \forall j \quad 1 \leq j \leq m \\ & x_i \in \{0,1\} \quad \forall i \quad 1 \leq i \leq n \end{aligned}$$

where:

- p_i is the profit of item i,
- x_i is the decision variable, it is set to 1 if the item is in the solution, 0 otherwise,
- r_{ij} is the coefficient or weight of item i in constraint j, and

b_j is the limit or maximum capacity of constraint j.

The multi-dimensional knapsack problem can be applied to problems such as the capital budgeting problem, the cargo loading problem, the cutting stock problem, and the processor allocation problem in distributed systems [288]. It is also widely used as a benchmark problem to compare new general-purpose meta-heuristics [61, 134, 241].

3.3.1.1 Implementation

A vector of n components gives the representation of the solution for the multi-dimensional knapsack problem. Each component represents an item where n is the number of items that could potentially be put in the knapsack. If the i^{th} item is in the knapsack then a value of 1 is assigned to its position in the solution vector, else a value of 0 is assigned to represent that this item is not in the knapsack. Figure 3.9 shows a representation for the multi-dimensional knapsack problem.

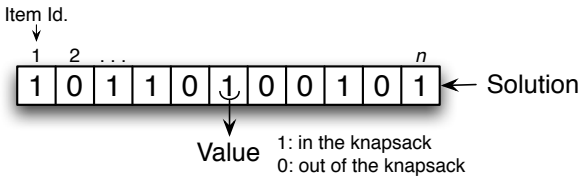


Figure 3.9: An example of the a solution vector for the MKP.

The initial solution is created through a random generation of the values $0, 1$ for each item. This entails that the initial solution could be feasible or infeasible, allowing the extremal optimisation process the task of moving an infeasible initial solution toward a feasible region. In the case that an initial feasible solution is generated, it is saved in S_{best} as the best solution found so far.

The fitness function evaluation follows the differentiated fitness evaluation scheme introduced in Section 3.2.1. Thus, the differentiated value for the multi-dimensional knapsack problem is given in Equation 3.3.

$$\lambda(x_i) = \begin{cases} -\left(S_i + \left(\frac{p_i}{p_{max}+1}\right)\right) & ,\forall i, x_i = 0 \\ \\ V_i + \left(\frac{p_i}{p_{max}+1}\right) & ,\forall i, x_i = 1 \end{cases} \tag{3.3}$$

where:

S_i is the remaining number of satisfied constraints when the i^{th} item is placed into the knapsack,

V_i is the remaining number of violated constraints when the i^{th} item is removed from the knapsack,

p_i is the profit of the i^{th} item, and

p_{max} is the maximum profit.

The fitness evaluation for this problem has the characteristic that it never takes out an item if the current solution is feasible, and never adds an item if the current solution is infeasible. This represents a considerable saving in runtime as only the relevant items to be analysed are considered.

When the solution is feasible, the components that potentially could be put in the knapsack, ensuring a high number of satisfied constraints and contributing a high profit, are considered as the components that degrade the current solution. This is because the non-inclusion of some of these items in the knapsack does not allow for obtaining a better evaluation of the solution. On the other hand, when the solution is infeasible, the components that potentially could be taken out from the knapsack, significantly reducing the number of violated constraints and having a low profit, are considered as the components that degrade the current infeasible solution. This is because its inclusion in the knapsack generates a higher level of infeasibility.

The roulette wheel technique (RWS) is used to select one of the worst evaluated items according to its probability P. In the knapsack problem, the replacement of the value for the chosen component (item) is a simple process because it only has to exchange its value from 0 to 1 or from 1 to 0 as appropriate.

When a feasible solution is found, then the secondary search mechanism is activated. The double traverse local search performs the swapping of items that are inside the knapsack with those that are outside it. Algorithm 9 describes the pseudocode of the local search implementation for the multi-dimensional knapsack problem.

The hybrid extremal optimisation procedure is repeated for a preset number of iterations. Finally, the best-found solution with its objective function evaluation are returned as outputs. Algorithm 10 shows the hybrid extremal optimisation pseudocode for the multi-dimensional knapsack problem.

Algorithm 9 The local search pseudocode for the MKP.

```
repeat
  for each item i within the knapsack do
    for each item j out of the knapsack do
      if the solution remains feasible when item i is exchanged by item j AND
      the evaluation of the new solution is increased then
        Implement the operation and update the variables involved in the action
      end if
    end for
  end for
  for each item i within the knapsack do
    for each item j and j' outside of the knapsack do
      if the solution remains feasible when item i is exchanged by item j and j'
      AND the evaluation of the new solution is increased then
        Implement the operation and update the variables involved in the action
      end if
    end for
  end for
until there is no improvement in the solution
```

Algorithm 10 The HEO framework for the MKP.

```
Generate the probability vector P
Initialise a solution S with random values from {0, 1}
if S feasible then S_best ← S
repeat
  if the current solution is feasible then
    Evaluate the fitness λ(x_i) using the first branch of Equation 3.3
  else
    Evaluate the fitness λ(x_i) using the second branch of Equation 3.3
  end if
  Rank the items according to their individual fitnesses λ(x_i) from the worst to
  the best
  Select an item based on the probability of its rank P using RWS
  Obtain a S_new in the neighbourhood of S changing the value of the selected
  item to 0 or 1 as appropriate
  Evaluate the new solution S_new
  if S_new is feasible then
    S_new = Apply the double traverse local search mechanism to S_new
    if the evaluation of S_new is better than the evaluation of S_best then
      S_best ← S_new
    end if
  end if
until the termination condition is satisfied
return S_best and the evaluation of S_best
```

3.3.2 Bin Packing Problem (BPP)

The bin packing problem is a constrained combinatorial problem with a theoretical and practical importance present in several processes in science, industry, and commerce. The bin packing problem is applicable to tasks such as the backup of tapes or disks of equal capacity, to allocate data onto blocks of memory with the same size, to assign processes on a system with identical parallel processors, or when products and/or resources have to be stored in containers. All these tasks follow the idea of minimising the wastage of resources in the bins [73, 72, 200].

There are many variations of this problem depending on the features and the limitation of the items to be stored and the objects that they must contain [109]. For example, production and distribution tasks require that a series of items with different shapes, sizes, or weights be packed into bins or boxes with either similar or different limited capacity. For this research, the traditional linear instance has been chosen because it is commonly used in practice [72]. Hence, a considerable number of relevant research work and benchmark tests can be found to validate the proposed approach.

The one-dimensional bin packing problem is the simplest version of this problem and can be formally described as follows. An infinite set of bins with the same bin capacity $C > 0$, and a finite set of n items with different weights in the range $0 < w_i \leq C$ is to be packed. The aim is to find the smallest number m of bins needed to contain all the n items. The combined weight of the items packed in each bin cannot exceed the bin capacity C, and each item can be assigned to one and only one bin.

3.3.2.1 Implementation

For the bin packing problem, the solution is composed of items of different weights that must be used to fill a number of bins. The objective is to minimise the number of bins without overloading them.

The representation of the solution for the bin packing problems is given by a vector where each component of the vector represents an item. The value of the component is the bin to which that item is assigned. With this representation, changes in the value of each component are expressed by either a movement or swapping of the items to other bins. Figure 3.10 gives a graphical representation for the bin packing problem.

The Best Fit Decreasing (BFD) strategy [202] is used to generate the initial solution. BFD sorts the items in decreasing order according to weight and then tries

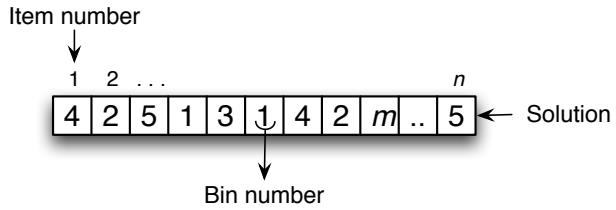


Figure 3.10: The bin packing problem representation using a vector of items, each containing the bin to which items are assigned. For example, item 1 is assigned to bin 4.

to put each item into a bin that has minimal free space. BFD guarantees to use no more than $\frac{11}{9}B + 1$ bins [202], where B is the optimum number of bins. The reason for generating an initial solution using BFD, rather than purely at random, is based on the interest in minimising the initial numbers of bins necessary to contain all the items making the main process more efficient. As the initial solution is always feasible, this is automatically assigned to S_{best}.

The differentiated fitness evaluation is applied to select one of the items that degrades the solution to be moved into another bin. Equation 3.4 illustrates the fitness evaluation for both cases based on the storage capacity of the bin and the weight of the item.

$$\lambda(x_i) = \begin{cases} -(C - \sum w_j) - \left(1 - \frac{w_i}{w_{max}+1}\right) & \forall j \in Bin_{solution[x_i]} \\ \\ (C - \sum w_j) + \left(\frac{w_i}{w_{max}+1}\right) & \forall j \in Bin_{solution[x_i]} \end{cases} \tag{3.4}$$

where:

C	is the capacity of the bin,
w_i	is the weight of the i^{th} item,
w_{max}	is the maximum item weight,
$\sum w_j$	is the sum of all j items in the bin assigned to item x_i, and
$Bin_{solution[x_i]}$	is the bin where the i^{th} item is assigned.

When the current solution is feasible, the items in the bins with large free spaces available are considered as components that degrade the solution since the aim of bin packing problems is to minimise the numbers of bins. However, when the current

solution is infeasible, the items in the most overloaded bins are considered as the components that degrade the solution.

The roulette wheel technique is used to select one of the worst evaluated items according to its probability P. In the bin packing problem, the replacement of the value for the chosen component is interpreted as the movement of one item from the assigned bin to another bin. First, a light item is chosen with a greater probability than a heavy item from the assigned bin, as it is less likely to overload another bin. Second, with the aim of performing a more efficient assignment strategy than the traditional random procedure, the extremal optimisation selection scheme is again applied to select the bin where the selected item is moved. Hence, the remaining available bins are ranked by a fitness value that is defined in Equation 3.5.

$$\lambda'(b_i) = \frac{R_i}{N_i} \qquad \forall i \text{ such that } 0 < R_i < C \tag{3.5}$$

where:

$\lambda'(b_i)$	is the fitness of the i^{th} bin,
R_i	is the current weight of the i^{th} bin, and
N_i	is the number of items in the i^{th} bin.

The idea of this selection process is to try to put the item in a bin where the relation $used\_space/number\_of\_items$ is the smallest. This means that between two bins with the same free space to receive a new item, it is preferable to use the bin with more items of smaller size. This is because if the bin gets overloaded then it is easier to adjust several small items than a few large items within the bin for the infeasible solution procedure and so obtain a feasible solution again. Algorithm 11 shows the procedure to choose the bin where the selected item will be assigned.

Algorithm 11 The replacement component procedure for the BPP to obtain the S_{new} in the neighbourhood of S.

1: Evaluate the fitness $\lambda'(b_j)$ for the remaining bins using Equation 3.5
2: Rank the bins according to their individual fitnesses $\lambda'(b_j)$ from the worst to the best
3: Select a bin based on the probability of its rank P using RWS
4: Put the chosen item in the selected bin b_i into the selected bin b_j

When a feasible solution is found then the secondary search mechanism is activated. For this problem, an improved tailor-made local search technique, that is based on the work developed by Falkenauer [112], which in turn was inspired by the work of Martello and Toth [203], was developed. The original local search mechanism was

applied to an ant colony approach by Levine and Ducatelle [188].

The secondary search mechanism begins by moving all the items contained inside the two least full bins into a (temporary) free bin. Four sequential steps are then applied for each non-full bin. These steps consist of interchanging one or two items between the current bin and the bin with free items so that the current bin can be refilled to the limit. The first step swaps two current items for two free items, the second step swaps two current items for one free item and the third step swaps one current item for one free item. A new fourth step that swaps one current item for two free items, is added to the algorithm. Next, the free items are reinserted into bins provided that these have enough space to contain them. Finally, the remaining items in the free bin are put into a new bin. This local search process is repeated while new solutions are feasible.

The hybrid extremal optimisation procedure is repeated for a preset number of iterations. Finally, the best-found solution and its objective function evaluation are returned as outputs. Algorithm 12 shows the hybrid extremal optimisation pseudocode for the bin packing problem.

Algorithm 12 The HEO framework for the BPP.

Generate the probability vector P
Initialise a feasible solution S using the BFD method
$S_{best} \leftarrow S$
repeat
  if the current solution is feasible **then**
    Evaluate the fitness $\lambda(x_i)$ using the first branch of Equation 3.4
  else
    Evaluate the fitness $\lambda(x_i)$ using the second branch of Equation 3.4
  end if
  Rank the items according to their individual fitnesses $\lambda(x_i)$ from the worst to the best
  Select an item based on the probability of its rank P using RWS and choose a light item from it.
  Obtain a S_{new} in the neighbourhood of S using the defined procedure in Algorithm 11
  Evaluate the new solution S_{new}
  if S_{new} is feasible **then**
    S_{new} = Apply the double traverse local search mechanism to S_{new}
    if the evaluation of S_{new} is better than evaluation of S_{best} **then**
      $S_{best} \leftarrow S_{new}$
    end if
  end if
until the termination condition is satisfied
return S_{best} and the evaluation of S_{best}

3.3.3 Generalised Assignment Problem (GAP)

The generalised assignment problem is a well-known $\mathcal{NP}$-hard [125] problem. This is considered as a fundamental constrained problem in combinatorial optimisation [48]. Like the knapsack and the bin packing problems, many real-world problems can be modelled by generalised assignment problems. Hence, it is possible to find a wide general usage of this in many applications [51, 52].

The problem consists of assigning n jobs to be processed by m agents with the objective of minimising the required cost to perform this operation. An agent may process one or more jobs. Each job must be assigned to one and only one agent. Each agent has a limited capacity to perform jobs and this capacity is not necessarily the same for all agents. The aim is to achieve the lowest overall cost assignment of jobs to the available agents bearing in mind the agent resource constraints. Formally, it can be represented as:

$$\begin{aligned}
\text{Minimise} \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij} \\
\text{Subject to} \quad & \sum_{i=1}^{n} a_{ij}x_{ij} \leq b_j \quad \forall j \quad 1 \leq j \leq m \\
& \sum_{j=1}^{m} x_{ij} = 1 \quad \forall i \quad 1 \leq i \leq n \\
& x_{ij} \in \{0,1\} \quad \forall i \quad 1 \leq i \leq n, \forall j \quad 1 \leq j \leq m
\end{aligned}$$

where:

- c_{ij} is the cost of assigning job i to agent j,
- a_{ij} is the resource required by agent j to perform job i,
- x_{ij} is the job's assignment, 1 if the job is assigned to agent j, 0 otherwise,
- b_j is the maximum capacity of agent j.

The generalised assignment problem can be applied to problems such as data storage and retrieval in disks [7], process assignments to computer networks [19], scheduling [51], resource allocation [22] and vehicle routing [17].

3.3.3.1 Implementation

The representation of the solution for the generalised assignment problem is given by a vector of n components. Each component represents a job where n is the number of jobs that must be assigned to one of the m available agents. If the i^{th} job is allocated

to the j^{th} agent then the value that identifies this agent is assigned to its position in the solution vector. Figure 3.11 shows a representation for the generalised assignment problem.

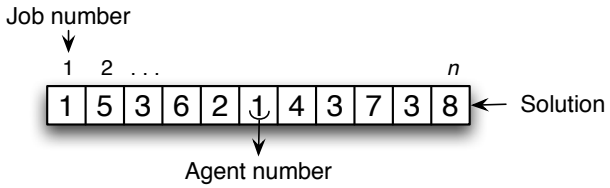


Figure 3.11: An example of a solution vector for the GAP.

The initial solution is created through a random generation of values in the range $(1, m)$ for each job. As in the multi-dimensional knapsack problem, the initial solution could be feasible or infeasible, so the extremal optimisation process attempts to transform an infeasible initial solution into a feasible one. In the case of an initial feasible solution, it is saved in S_{best} as the best solution found so far.

The differentiated fitness evaluation scheme is applied to calculate the fitness for each job. This evaluation will assess whether the current allocation of the i^{th} job degrades the solution. Equation 3.6 states the fitness function for the generalised assignment problem as follows.

$$\lambda(x_i) = \begin{cases} -\sum_{j=1, j\neq x_i}^{m} \left(S_i + \left(1 - \frac{c_{ij}}{c_{max}+1} \right) \right) & , \forall i \\ \\ \sum_{j=1, j\neq x_i}^{m} \left(V_i + \left(1 - \frac{c_{ij}}{c_{max}+1} \right) \right) & , \forall i \in \tilde{M} \end{cases} \tag{3.6}$$

where:

- S_i is 1 if the j^{th} agent is not overloaded when the i^{th} job is moved to it, otherwise it is 0,
- V_i is 1 if the j^{th} agent is overloaded when the i^{th} job is moved to it. It is set to 2 if the j^{th} agent where the i^{th} job is moved to is already overloaded, otherwise it is 0,
- c_{ij} is the cost of assigning the i^{th} job to the j^{th} agent,
- c_{max} is the assignment cost with the highest value, and
- $\tilde{M}$ is the set of overloaded agents.

When the solution is feasible, the jobs that potentially could be assigned to any

other agent, without overloading a high number of agents with a reduced allocation cost, are considered as the components that degrade the present solution. This is because its current assignment prevents the attainment of a better evaluation. On the other hand, only jobs that have been allocated to an overloaded agent are considered. From these jobs, those that potentially could be assigned to any other agent, without overloading a large number of agents with a reduced allocation cost, are considered as the components that degrade the present solution. This is because its current assignment generates a greater level of infeasibility.

The roulette wheel technique is used to select one of the worst evaluated jobs according to its probability P. In the generalised assignment problem, the replacement of the value for the chosen component (job) is carried out by a random selection of an agent that must be different to the current agent assigned to the chosen job.

When a feasible solution is found then the secondary search mechanism is activated. For the generalised assignment problem, the double traverse local search is applied by carrying out swaps between the jobs that are allocated to different agents. The local search pseudocode that has been adapted to work with the generalised assignment problem is described in Algorithm 13.

Algorithm 13 The local search pseudocode for the GAP.

```
1:  repeat
2:    for each job i do
3:      for each job i' do
4:        if job i and job i' are assigned to different agents AND the solution remains
          feasible when job i is exchanged by job i' AND a better solution is received
          then
5:          Implement the operation and update the variables involved in the action
6:        end if
7:      end for
8:    end for
9:    for each job i do
10:     for each job i' and i'' do
11:       if job i and jobs i' and i'' are assigned to different agents AND the solution
          remains feasible when job i is exchanged by jobs i' and i'' AND a better
          solution is received then
12:         Implement the operation and update the variables involved in the action
13:       end if
14:     end for
15:   end for
16: until there is no improvement in the solution
```

The hybrid extremal optimisation procedure is repeated for a preset number of

iterations. Finally, the best-found solution and its objective function evaluation are returned as outputs. Algorithm 14 shows the hybrid extremal optimisation pseudocode for the generalised assignment problem.

Algorithm 14 The HEO framework for the generalised assignment problem.

Generate the probability vector P
Initialise a solution S with random integer values in the range $(1, m)$
if S feasible **then** $S_{best} \leftarrow S$
repeat
  if the current solution is feasible **then**
    Evaluate the fitness $\lambda(x_i)$ using the first branch of Equation 3.6
  else
    Evaluate the fitness $\lambda(x_i)$ using the second branch of Equation 3.6
  end if
  Rank the jobs according to their individual fitnesses $\lambda(x_i)$ from the worst to the best
  Select a job based on the probability of its rank P using RWS
  Obtain S_{new} in the neighbourhood of S by changing the value of the selected job to a random agent other than the one already assigned.
  Evaluate the new solution S_{new}
  if S_{new} is feasible **then**
    S_{new} = Apply the double traverse local search mechanism to S_{new}
    if the evaluation of S_{new} better than the evaluation of S_{best} **then**
      $S_{best} \leftarrow S_{new}$
    end if
  end if
until the termination condition is satisfied
return S_{best} and the evaluation of S_{best}

3.3.4 Possible Application to Other Capacitated Combinatorial Problems

The characteristic of capacitated combinatorial problems is that they possess some resources with a limited capacity that must be handled so that these limitations are not violated. Figure 3.1 shows some of them in addition to the above described multi-dimensional knapsack problem, bin packing problem and generalised assignment problem. The latter broadly representative of this sort of problem. Next, a concise explanation of how HEO can by applied to the frequency assignment problem [264], the single source capacitated facility location problem [228], the capacitated minimal spanning tree problem [12], the capacitated vehicle routing problem [84] and the capacitated set covering problem [53] is given.

3.3.4.1 Frequency Assignment Problem (FAP)

In last few decades, the discipline of wireless communication has experienced large growth, increasing consumer demand and technological innovation. There has been a correspondingly intensive usage of the limited radio spectrum by different applications [173]. With the increasing use of mobile telephones, Wi-Fi networks, GPS systems combined with the existing and growing employment of radio and television broadcasting, the efficient assignment of increasingly scarce frequencies is needed.

The frequency assignment problem aims to select a subset of frequencies that minimises the interference among transmitters and receivers according to the different requirements or restrictions imposed on the particular problem to be solved. Some of these requirements are to minimise either the number of used frequencies or the largest assigned frequency in order to leave as much free space as possible in frequency bands for future requirements.

In the frequency assignment problem, a number of n transmitters defined in the set T have to be assigned a limited number of m frequencies from the set F of available frequencies. The frequency assignment must take into consideration the interference limitations between frequencies. Let d_{ij} be the minimum allowed distance for the frequencies F_i and F_j that have to fulfil the relation $|F_i - F_j| > d_{ij}$. According to the frequency spectrum, interference can manifest when nearby frequencies are assigned to two transmitters that are close to each other.

The solution for the FAP can be represented through a vector of n elements where each component of the vector is associated with a transmitter. For every transmitter the vector stores the frequency that was assigned to it. Figure 3.12 shows a representation for the frequency assignment problem.

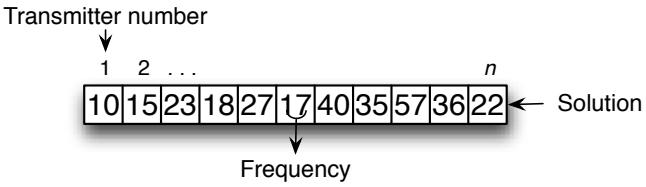


Figure 3.12: An example of a solution vector for the FAP.

The initial solution for the frequency assignment problem is generated by a random assignment from the valid frequencies for each transmitter.

The differentiated fitness evaluation for the FAP is presented in Equation 3.7. The fitness evaluates the contribution to the optimality or violation for each component of

the solution.

$$\lambda(x_i) = \begin{cases} \pm S_i & \forall i \\ \\ \max\{|F_i - F_j| - d_{ij}\} & \forall i, j \in \tilde{M} \end{cases} \tag{3.7}$$

where:

- S_i is either the number of times the frequency assigned to the i^{th} transmitter has been allocated or the frequency value assigned to the i^{th} transmitter according to the objective function to be fulfilled,
- F_i is the frequency assigned to the i^{th} transmitter,
- d_{ij} is the minimum allowed distance for the frequencies F_i and F_j and
- $\tilde{M}$ is the set of transmitters with frequency conflicts.

When the solution is feasible, the fitness evaluation depends on the defined objective function. If the objective is to minimise the number of used frequencies then transmitters that have assigned frequencies which have been allocated fewer times are the components that degrade the solution. This is because these transmitters could use frequencies more repeatedly allocated so that frequencies which are less used can become available for future use. On the other hand, if the objective is to minimise the largest assigned frequency then transmitters that have the highest assigned frequencies are the components that degrade the solution. This is because their current frequency assignment prevents a better evaluation from being obtained. Now, when the solution is infeasible, only transmitters that have assigned a frequency in conflict with some other transmitter are considered. From these transmitters, those that have assigned a frequency that generate the large constraint violation are considered as the components that degrade the present solution. This is because its current assignment generates a greater level of infeasibility.

The replacement of the value for the chosen component (transmitter) is carried out by a random assignment from the valid frequencies for each transmitter that must be different to the current one. When a feasible solution is found then the double traverse local search is activated. For this problem, the local search mechanism is performed by swaps between the frequencies that are assigned to different transmitters.

3.3.4.2 Single Source Capacitated Facility Location Problem (SSCFLP)

The SSCFLP is a problem that can be applied in areas such as distribution systems planning, telecommunication networks design and logistics [10]. This problem has

the aim of locating a number m of potential facilities at some predefined sites that must satisfy the demand of products for a group of n customers with the objective of minimising the transportation cost tc_{ij} between facilities and customers. The cost includes a fixed charge f_j that is associated with the opening of the j^{th} facility. For each customer i there is a specific demand d_i that must be supplied by a single facility that has a capacity c_j.

The solution for the SSCFLP can be represented through a vector of n elements where each component of the vector is associated with each customer. For every customer, the vector stores the facility that was selected to satisfy the customer's demand. Figure 3.13 shows a representation for the single source capacitated facility location problem.

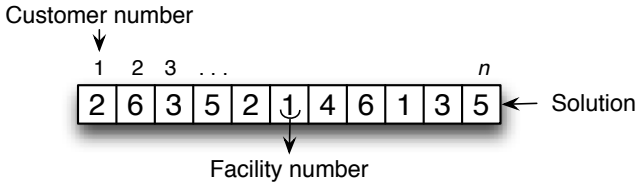


Figure 3.13: An example of a solution vector for the SSCFLP.

The initial solution for the SSCFLP is generated by a random location of the facilities assigned to supply the demand of the n customers.

The differentiated fitness evaluation for the SSCFLP is presented in Equation 3.8. The fitness evaluates the contribution to the optimality or violation for each node of the solution.

$$\lambda(x_i) = \begin{cases} -\sum_{j=1, j\neq x_i}^{m} \left(S_i + \left(1 - \frac{tc_{ij}}{tc_{max}+1} \right) \right) & , \forall i \\ \\ \sum_{j=1, j\neq x_i}^{m} \left(V_i + \left(1 - \frac{tc_{ij}}{tc_{max}+1} \right) \right) & , \forall i \in \tilde{M} \end{cases} \tag{3.8}$$

where:

- S_i is 1 if the j^{th} facility is not overloaded when the i^{th} customer satisfies its demand from it, otherwise it is 0,
- V_i is 1 if the j^{th} facility is overloaded when the i^{th} customer satisfies its demand from it. It is 2 if the j^{th} facility is already overloaded, otherwise it is 0,

tc_{ij} is the transportation cost from the i^{th} customer to the j^{th} facility,
tc_{max} is the maximum transportation cost, and
$\tilde{M}$ is the set of overloaded facilities.

When the solution is feasible, the customers that potentially could satisfy its demand from any other facility location, without generating an excess demand for the facility and with a reduced transportation cost, are considered as the components that degrade the present solution. This is because its current facility assignment prevents a better evaluation from being obtained. On the other hand, only customers who have been assigned to facilities that have exceeded their capacity, are considered. From these customers, those that potentially could be reassigned to any other facility without exceeding a large number of them and with a reduced transportation cost, are considered as the components that degrade the present solution. This is because its current assignment generates a greater level of infeasibility.

The replacement of the value for the chosen component (customer) is carried out by a random selection of a facility that must be different to the current one. When a feasible solution is found, then the double traverse local search is activated. For this problem, the local search mechanism is performed by swaps between the customers who are assigned to different facilities.

3.3.4.3 Capacitated Minimal Spanning Tree Problem (CMSTP)

The CMSTP is generally applied to create efficient configurations for centralised communication networks, but it has also been investigated for its relevance in other practical applications [12, 204]. The objective in the CMSTP is to find the minimum cost tree through the extension of a set of nodes that must satisfy a set of capacity constraints associated with the edges of the tree. This problem consists of n nodes where the node 0 represents the root of the tree, d_i represents the traffic demand of the i^{th} node and c_{ij} represents the cost of establishing the connection between the i^{th} and the j^{th} nodes. The aim of the CMSTP is to find a tree so that the traffic on each subtree connected to the root is less than or equal to Q and the total connection cost is minimised.

The solution for the CMSTP can be represented through a vector of n elements where each component of the vector is associated with each node of the tree and the root of the tree is identified with the value 0. For every node, the vector stores the parent node for the i^{th} node in the tree. Figure 3.14 shows a representation for the capacitated minimal spanning tree problem.

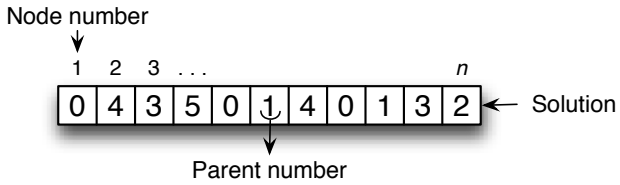


Figure 3.14: An example of a solution vector for the CMSTP.

The initial solution for the CMSTP can be generated both by connecting every node into the root node or by carrying out random connections of them. The former produces a feasible solution but with a likely high connection cost, and the latter may generate an infeasible solution.

The differentiated fitness evaluation for the CMSTP is presented in Equation 3.9. The fitness evaluates the contribution to the optimality or violation for each node of the solution.

$$\lambda(x_i) = \begin{cases} -\sum_{j=1, j\neq x_i}^{n} \left(S_i + \left(1 - \frac{c_{ij}}{c_{max}+1} \right) \right) & , \forall i \\ \\ \sum_{j=1, j\neq x_i}^{n} \left(V_i + \left(1 - \frac{c_{ij}}{c_{max}+1} \right) \right) & , \forall i \in \tilde{M} \end{cases} \tag{3.9}$$

where:

S_i	is 1 if the j^{th} subtree is not overloaded when the i^{th} node is moved to it, otherwise it is 0,
V_i	is 1 if the j^{th} subtree is overloaded when the i^{th} node is moved to it. It is 2 if the j^{th} agent where the i^{th} job is moved to is already overloaded, otherwise it is 0,
c_{ij}	is the connection cost from the i^{th} node to the j^{th} node,
c_{max}	is the maximum connection cost, and
$\tilde{M}$	is the set of overloaded subtrees

.

When the solution is feasible, the nodes that potentially could be assigned to any other subtree, without overloading the traffic demand for the subtree with a reduced connection cost, are considered as the components that degrade the present solution. This is because its current connection prevents it from obtaining a better evaluation. On the other hand, only nodes that have been connected to an overloaded subtree are considered. When one of these nodes is reassigned to each of the other subtrees and as result no subtree is overloaded or a small number of the subtrees are overloaded, then this node is considered as a component that degrades the current solution. If

two or more nodes have the same number of overloaded subtrees then the connection cost is used to differentiate which of them degrade the solution in a greater degree. From them, nodes with reduced connection costs are more likely to be selected. Thus, the current connection of the nodes that potentially can reduce both the number of overloaded subtrees and the connection cost generate a greater level of infeasibility.

The replacement of the value for the chosen component (node) is carried out by a random selection of a parent node that must be different to the current parent. When a feasible solution is found then the double traverse local search is activated. For this problem, the local search mechanism is applied by carrying out swaps between the nodes that are connected to different parents.

3.3.4.4 Capacitated Vehicle Routing Problem (CVRP)

Many logistic situations must deal with the problem of routing a set of elements from a source to a group of destinations that are geographically dispersed. Some of the entities that face this sort of problem are: courier and postal services, local trucking companies, demand responsive transportation services, department store delivery services and food distribution companies [279], just to name a few.

The capacitated vehicle routing problem describes the situation when a central depot must satisfy the single product demand d_i for a set of n customers through m potential different routes. Each route uses a vehicle and all required vehicles have the same capacity C. There exists a transportation cost tc_{ij} that indicates the required transportation time between two points of the route. All routes start and end at the depot. The objective is to minimise the number of routes and the sum of the transportation time. The total demand of the single product for each route may not exceed the capacity of the vehicle.

The solution for the CRVP can be represented through an array of n elements where each component of the array has an associated customer and the route that will be used to supply the respective customer. Figure 3.15 shows a representation for the capacitated vehicle routing problem.

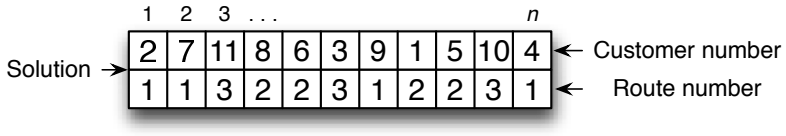


Figure 3.15: An example of a solution vector for the CVRP.

The route itinerary is read from left to right in the solution array. For example, the

solution in Figure 3.15 generates the following result. The vehicle for route number 1 departs from the depot and visits customers 2, 7, 9 and 4 and then it returns to the depot. Route number 2 departs and visits customers 8, 6, 1 and 5 and then it returns to the depot. Route number 3 visits customers 11, 3 and 10 and then it returns to the depot.

The CRVP can been seen as a generalisation of the traveling salesman problem, for the generation of the sequence of the routes, and the bin packing problem, for the customer distribution in the least numbers of routes. Hence, some ideas applied to solve these two problems could be used to find a solution in the CRVP.

The initial solution for the CVRP is generated using the best fit decreasing strategy in a similar way as was previously performed for the bin packing problem. This generates a feasible initial solution with as few routes as possible.

The differentiated fitness evaluation for the CVRP is presented in Equations 3.10, 3.11 and 3.12. The fitness evaluates the contribution to the optimality or violation for each node of the solution.

$$\lambda(x_i) = \begin{cases} -\left(C - \sum_{j \in route[x_i]} d_j\right) + \sum_{k=1, k \neq route[x_i]}^{m} S_k + \left(\frac{\overline{tc}_i}{\overline{tc}_{max}+1}\right) \\ \\ \left(C - \sum_{j \in route[x_i]} d_j\right) + \left(\frac{d_i}{d_{max}+1}\right) \end{cases} \tag{3.10}$$

$$\overline{tc}_i = \sum_{l=1, l \neq customer[x_i]}^{n} tc_{il} \tag{3.11}$$

$$\overline{tc}_{max} = \max\{\overline{tc}_i | i = 1, \ldots, n\} \tag{3.12}$$

where:

C	is the capacity of the vehicle,
d_i	is the demand of the i^{th} customer,
d_{max}	is the maximum demand of all customers,
$\sum d_j$	is the sum of all j demands in the route assigned to customer x_i,
S_k	is 1 if the k^{th} route is not overloaded when the i^{th} customer is moved to it, otherwise it is 0,
tc_{il}	is the transportation cost from the i^{th} to the l^{th} customer,
$\overline{tc}_i$	is the transportation cost from the i^{th} customer to all other customers, and
$\overline{tc}_{max}$	is the maximum $\overline{tc}$ value for all customers.

When the solution is feasible, the components that degrade the solution are ranked, taking into consideration three criteria. First, the customers that are in the routes with a large free available capacity because they could be in other routes and thereby one of those routes could be eliminated. To this criterion is added another, which categorises those customers that, despite being moved to the other routes, still maintain a high degree of optimality in the solution. Finally, the third included criterion assesses if the movement of the customer to other routes generates a decrease in transportation costs. Thus, the customers that fulfil all the criteria are considered as components that degrade the solution. This is because the objective is to minimise the numbers of routes with the minimum transportation cost. However, when the current solution is infeasible, the customers in the most overloaded routes are considered as the components that degrade the solution.

The replacement procedure contains two steps. First, the route assigned to the selected component is changed to a random one, within the range of possible values. This new route could be either a different route or the same route that was previously assigned. Second, a swap of the information associated with the selected component and the other component randomly selected is performed. Figure 3.16 illustrates the replacement procedure.

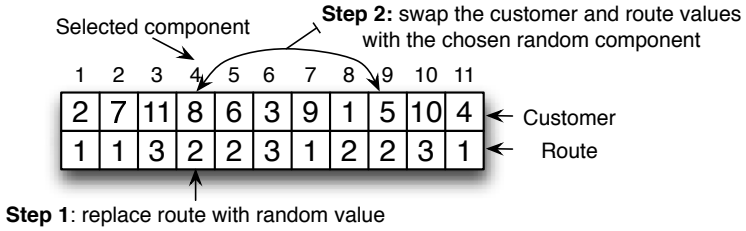


Figure 3.16: The replacement procedure for the capacitated vehicle routing problem.

When a feasible solution is found then the secondary search mechanism is activated. For this problem, the improved tailor-made local search technique proposed for the bin packing problem and the double traverse local search can be applied in a memetic way, as explained in Section 3.2.2.

3.3.4.5 Capacitated Set Covering Problem (CSCP)

The set covering problem refers to those problems that require coverage of a range of necessities at the lowest possible cost. In other words, it is a problem of selecting a number of sets $S' \subseteq S$ that make it possible to contain all elements involved in the

problem with the objective of generating the lowest possible cost associated with the selection of the sets. Additionally, for the capacitated case, the condition that each element has a demand that can be satisfied by one or more sets, which in turn have an amount of supplies to cover part of the demand for an element, must be incorporated.

The CSCP has been applied to solve problems such as minimising the number of directional antennas [26], looking for the minimum cost subgraph in steiner trees [20], and in some particular variants of the scheduling problems [21].

Formally, the CSCP has a number of n elements with a demand d_j and m sets with a limited amount of supplies s_i. Each set has associated a cost c_i when it is included in the solution. The allocation of elements into sets is registered in the matrix a_{ij}. The objective is to select a collection of sets that cover all n elements with the least cost, without violating any demand requirement.

The solution for the CSCP can be represented through an array of m sets where each component of the array indicates, in a binary value (1 or 0), whether or not the respective set is present in the solution. Figure 3.17 shows a representation for the capacitated set covering problem.

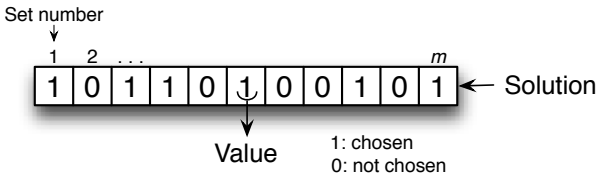


Figure 3.17: An example of a solution vector for the CSCP.

The initial solution for the capacitated set covering problem is obtained by either a random selection of sets or a selection that ensures all elements are covered. The former could generate an infeasible solution due to not all elements being covered and/or the demands for some elements are not satisfied. The latter could generate an infeasible solution only because the demands for some elements are not satisfied. For both cases, if the i^{th} set is selected then a value of 1 is assigned, 0 otherwise.

The differentiated fitness evaluation for the CSCP is presented in Equations 3.13, 3.14 and 3.15. The fitness evaluates the contribution to the optimality or violation for each component of the solution.

$$\lambda(x_i) = \begin{cases} \lambda'(x_i) & \text{if the solution is feasible} \\ \\ \lambda''(x_i) & \text{if the solution is infeasible} \end{cases} \tag{3.13}$$

$$\lambda'(x_i) = \begin{cases} -S_i c_i & ,\forall i \forall j : x_i, x_j = 1 \quad ,\text{if } \exists x_i \forall j : x_i \prec x_j, i \neq j \\ -\left(\sum_{j=1, j\neq i}^{m} S'_j + \frac{c_i}{c_{max}+1}\right) & ,\forall i \forall j : x_i, x_j = 0 \quad ,\text{if } \nexists x_i \forall j : x_i \prec x_j, i \neq j \end{cases} \tag{3.14}$$

$$\lambda''(x_i) = \begin{cases} -\left(V_i + \left(1 - \frac{c_i}{c_{max}+1}\right)\right) & ,\forall i : x_i = 0 \quad ,\text{if } \exists k : \sum_j^m a_{jk} x_j = 0 \\ -\left(V'_i + \left(1 - \frac{c_i}{c_{max}+1}\right)\right) & ,\forall i : x_i = 0 \quad ,\text{if } \exists k : \sum_j^m a_{jk} s_j x_j < d_k \end{cases} \tag{3.15}$$

where:

S_i	is 1 if $x_i \prec x_j$, i.e., set x_i dominates any other set x_j in the solution,
S'_j	is 1 if the i^{th} set does not generate a lack of demand for any element when this is swapped with the j^{th} set, otherwise it is 0,
V_i	is the number of uncovered elements that the i^{th} set could cover,
V''_i	is the number of elements with unsatisfied demand that the i^{th} set could satisfy,
c_i	is the cost of the i^{th} item,
c_{max}	is the maximum cost of all sets,
a_{jk}	is 1 if the j^{th} set covers the k^{th} element, 0 otherwise,
s_j	is the supply of the j^{th} set and
d_k	is the demand of the k^{th} element.

When the solution is feasible, two cases are analysed. First, if there exists at least one set that dominates another, from the selected sets in the current solution, then the sets that can be eliminated from the solution without leaving an uncovered element are considered as the components that degrade the current solution. This is because its current selection prevents it from obtaining a lower cost evaluation. Second, if none of the sets in the current solution dominate any other, that means that no set can be unassigned without leaving an element uncovered. In this case, the sets that potentially could be replaced by any other unassigned set, maintaining the feasibility of the solution, are considered as the components that degrade the present solution. This is because the selection of an alternative set could obtain a lower cost evaluation.

On the other hand, when the solution is infeasible, two cases are also analysed. First, if there exists at least one uncovered element then the sets that potentially could cover it are considered as the components that degrade the current solution. This is because these sets, which are not currently selected in the solution, generate a greater level of infeasibility and with the selection of one of them, this level could

be reduced or potentially eliminated. Second, if there exists at least one element with an unsatisfied demand then the sets that potentially could complete the remaining demand for these elements are considered as the components that degrade the current solution. This is because these sets, which are not currently selected in the solution, generate a greater level of infeasibility and with the selection of one of them, this level could be reduced or potentially eliminated.

The replacement of the value for the chosen component (set) is carried out by either a switch from 0 to 1 or from 1 to 0 randomly, as appropriate. When a feasible solution is found then the double traverse local search is activated. For this problem, the local search mechanism swaps sets that are in the solution with those that are not included in the solution.

3.3.5 Discussion

In this section the hybrid extremal optimisation framework has been applied to a set of eight capacitated and constrained combinatorial optimisation problems. For the first three problems; multi-dimensional knapsack, bin packing and generalised assignment, a thorough description of how the framework is implemented for each of them was elaborated. Once the applicability of the framework was tested, for the remaining five problems; frequency assignment, single source capacitated facility location, capacitated minimal spanning tree, capacitated vehicle routing and capacitated set cover, a concise description where the significant aspects of how the framework is applied to them was performed.

For each one of these problems, it appeared possible to apply the following main features of the hybrid extremal optimisation framework:

- The vector solution structure to represent the solution a their components.
- The differentiated fitness evaluation scheme to evaluate feasible and infeasible solutions through the contribution of their components.
- The secondary search mechanism to perform the fine grain search using the double traverse local search technique.

Thus, as evidenced by these examples, the hybrid extremal optimisation framework is very general and applicable to a range of these types of problems.

The next section performs the computational experiments for the problems of the multi-dimensional knapsack, bin packing and generalised assignment. This experimental study will allow to observe the efficiency, the effectiveness and the competitiveness of the hybrid extremal optimisation framework.

3.4 Computational Experiments

The proposed hybrid extremal optimisation algorithm was coded in the C language and compiled with `gcc`. The computing platform used to perform the tests had a 1.86 GHz Intel Core2 CPU and 917 MB of memory, running under Linux.

The sets of test data used for the problems came from the OR-Library [24]. Each problem instance was run ten times, and for each run, the random seed was changed to a new value.

All results are presented as a percentage gap that is determined using the theoretical optimal solution and the solution obtained by each analysed method. It is defined as $\%gap = \frac{b-a}{b} \times 100$, where a is the value of the obtained solution and b is the value of the theoretical optimal solution. A $\%gap$ of 0 means that the method found the optimal or best-known value.

The τ value is tested within the global range from 1.1 to 2.9 in increments of 0.1; even though the actual τ test range it is independently defined by each test data problem in the subsection of results. This is performed in order to select the best τ value to obtain efficient solutions and to compare the chosen value with those used in other research where the extremal optimisation heuristic has been applied. For example, Figure 3.18 illustrates the plotted graph result showing the average percentage gap for τ values tested within the range from 1.8 to 2.8 for the large test data for the MKP.

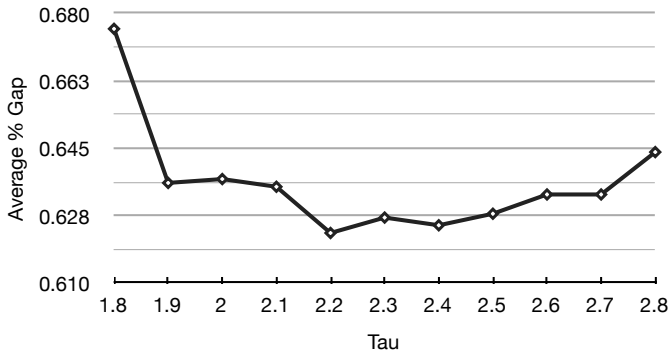


Figure 3.18: Result for the τ test for the large MKP problem instances where it can be observed that values around 2.2 work very well. This graph is illustrative of the performance curve for different tested values of τ, regardless of the test problem and the τ test range.

3.4.1 Small Test Data for the MKP

With the purpose of developing a first test with the hybrid extremal optimisation approach, a group of six small problems created and solved by Petersen [238] was used. The small problems' names are presented through the nomenclature MKNAP-P where P is the identification number of the problem. These problems are configured with a different number of constraints $m \in \{5, 10\}$ and number of items $n \in \{6, 20, 28, 39, 50, 60\}$ as shown in Table 3.12.

Problem name	n	m
MKNAP-1	06	10
MKNAP-2	20	10
MKNAP-3	28	10
MKNAP-4	39	05
MKNAP-5	50	05
MKNAP-6	60	05

Table 3.12: The features of the six small multi-dimensional knapsack problems.

These six problems were the same used by Randall and Lewis [245]. They implemented the first extremal optimisation approach to solve the multi-dimensional knapsack problem.

3.4.1.1 Results

Following the work of Randall and Lewis [245], the number of iterations was set to 500000 to run HEO under similar conditions. However, the τ value used by these researchers was not specified explicitly. For this reason, the small problems were run for values of τ from 1.1 to 1.7 in increments of 0.1. For values of τ out of this range, the results obtained were further away from the optimal value. The value of τ that obtained the best result was 1.4.

The results generated by the HEO algorithm were compared with those reported by the Standard Extremal Optimisation algorithm (SEO) and the Extended Extremal Optimisation Model (EEO) proposed by Randall and Lewis [245]. The EEO model is based on the Evolutionary Programming using Self Organised Criticality (EPSOC) meta-heuristic proposed by Lewis [190].

The mean and best results obtained with the number of iterations necessary to reach these results are shown in Table 3.13.

Problem				SEO				EEO	HEO			
Name	n	m	Opt.	%gap best result	Num. of iter.	%gap mean result	Num. of iter.	%gap best result	%gap best result	Num. of iter.	%gap mean result	Num. of iter.
MKNAP-1	06	10	3800	**0.00**	9140	**0.00**	9140	**0.00**	**0.00**	**4**	**0.00**	**6**
MKNAP-2	20	10	6120	1.31	166304	2.29	70808	1.31	**0.00**	**177**	**0.00**	**6880**
MKNAP-3	28	10	12400	1.77	161089	1.77	161089	1.53	**0.00**	**17738**	**0.00**	**52598**
MKNAP-4	39	05	10618	14.46	287120	15.03	197651	5.33	**0.00**	**1427**	**0.13**	**62698**
MKNAP-5	50	05	6339	3.82	172585	5.30	31215	3.66	**0.00**	**140**	**0.00**	**690**
MKNAP-6	60	05	6954	4.37	119523	5.34	169988	5.52	**0.00**	**6876**	**0.00**	**15498**

Table 3.13: A comparative table of test results for different EO models.

3.4.1.2 Discussion

Initially, the SEO algorithm found only one optimal value for the problem MKNAP-1. The results obtained for the remaining instances were close to the optimal values. The time taken to find the best and mean result was measured by the number of iterations required to reach it. After that, the Extended Extremal Optimisation Model (EEO) proposed by Randall and Lewis [245] was applied to determine if better results could be obtained. The results for the problems MKNAP-3 to MKNAP-5 were better but these did not reach the optimal value, and for the problem MKNAP-6 the result was slightly worse.

The results obtained for the proposed HEO method show that it found the optimal solution for each test problem. However, the optimal value for the test problem MKNAP-4 was not always found. The mean percentage gap for this particular test problem was 0.13%, which was better than the other techniques.

The results obtained showed that the proposed HEO method outperforms the previous work, finding the optimal solution for each small test problem. Thus HEO has proved to be a more efficient mechanism to solve the multi-dimensional knapsack problem than the previous extremal optimisation approaches.

3.4.2 Large Test Data for the MKP

As the results on the previous test were promising, HEO was applied to solve 270 large problems created and solved by Chu and Beasley [61].

The large problems are grouped by the number of constraints $m \in \{5, 10, 30\}$ and the number of items $n \in \{100, 250, 500\}$ making up nine groups of 30 problems each. These 30 problems are subdivided, at the same time, into three subgroups of ten problems by tightness ratio $\alpha \in \{0.25, 0.5, 0.75\}$ of the resource constraints, which was set using $b_j = \alpha \sum_{i=1}^{n} r_{ij}$.

3.4.2.1 Results

The number of iterations again was set to 500000. Unlike the small test data, for the large test data, the τ value was calculated through an empirical test with a reduced number of 100000 iterations and only three runs for each problem. This specific set of conditions to find the best τ value was due to the considerable computational time necessary to perform a full test for twenty or more different values of τ. The test mechanism consisted of measuring the result of the average percentage gap obtained by the LP relaxation technique [1] and HEO, that is, $\%gap = 100\left(\frac{opt^{LP} - opt^{HEO}}{/opt^{LP}}\right)$. Thus the τ value with the least percentage gap was chosen. τ was tested within the range of values between 1.8 and 2.8. For values outside this range of τ, the results were worse. The value selected for τ was 2.2.

Since the optimal solution values for this set of test problems were not known, the results were compared with those obtained by Chu and Beasley (CB-GA) [61], Raild (R-GA) [241], Uyar and Eryigit (UE-GA) [287], Gottlieb (G-EA) [134], and Vasquez, Hao and Vimont (VH-TS) [288, 295]. The average percentage gap with respect to the LP relaxed optimum was used to compare the results obtained for each method. Table 3.14 shows the values found and reported by each method with the average of the ten best values found for each tightness group and/or the average percentage gap with respect to the Linear Programming (LP) method [2].

3.4.2.2 Discussion

The results obtained show that HEO found the same values for the first group with five constraints and 100 items and similar values for the seventh group with 30 constraints and 100 items. For the latter, there was an improvement for the first subgroup of 0.01%; however, the second subgroup could not reach the same performance by a factor of 0.02%. For the remaining groups, the values were close to those previously obtained by other more complex, and less generalisable, methods. The CB-GA method uses a repair operator that works with a surrogate relaxation technique. R-GA is based on a pre-optimisation of the initial population through an LP relaxation technique, a repair operator, and a local optimisation operator. The UE-GA method uses a gene-based adaptive mutation approach and six parameters that must be set. The G-EA method uses a decoder technique based on a permutation representation for the crossover and mutation operator and five parameters must be set. The VV-TS combines LP relaxation with tabu search, which is used as a local search mechanism.

[1] These results were obtained by Chu and Beasley [61].

[2] Blank spaces are present because that information was not provided in the paper.

Problem			LP	CB-GA	R-GA	UE-GA	G-EA	VV-TS	HEO
			Average optimal	Average %	Average %	Average %	Average %	Average %	Average %
m	n	α	values	gap	gap	gap	gap	gap	gap
5	100	0.25	24438.39	**0.99**					**0.99**
		0.50	43449.50	**0.45**					**0.45**
		0.75	60663.76	**0.32**					**0.32**
Average				**0.59**	**0.59**		**0.59**		**0.59**
5	250	0.25	60547.41	**0.23**				**0.23**	0.55
		0.50	109411.71	0.12				**0.11**	0.25
		0.75	151676.33	**0.08**				**0.08**	0.13
Average				**0.14**	0.15		0.17	**0.14**	0.31
5	500	0.25	120717.02	0.09		0.59		**0.07**	0.64
		0.50	219595.79	**0.04**		0.21		**0.04**	0.27
		0.75	302434.90	0.03		0.09		**0.02**	0.14
Average				0.05	**0.04**	0.30	0.10	**0.04**	0.36
10	100	0.25	22960.49	**1.56**					1.66
		0.50	43000.79	**0.79**					0.88
		0.75	59844.23	**0.48**					0.50
Average				**0.94**	0.95		0.97		1.01
10	250	0.25	59290.15	0.51				**0.45**	1.31
		0.50	108980.95	0.25				**0.23**	0.56
		0.75	151560.07	0.15				**0.14**	0.30
Average				0.30	0.29		0.38	**0.27**	0.74
10	500	0.25	118835.77	0.24		0.60		**0.17**	1.21
		0.50	217503.76	0.11		0.27		**0.08**	0.46
		0.75	302775.74	0.07		0.15		**0.06**	0.25
Average				0.14	0.11	0.34	0.27	**0.10**	0.64
30	100	0.25	22305.35	2.91					**2.90**
		0.50	41994.84	**1.34**					1.36
		0.75	59693.58	**0.83**					**0.83**
Average				**1.70**	1.71		1.74		**1.70**
30	250	0.25	57554.09	1.19				**1.08**	1.84
		0.50	107229.81	0.53				**0.48**	0.80
		0.75	150903.67	0.31				**0.28**	0.41
Average				0.68	0.64		0.85	**0.61**	1.02
30	500	0.25	116184.37	0.61		0.97		**0.48**	1.47
		0.50	216729.77	0.26		0.43		**0.21**	0.73
		0.75	302855.69	0.17		0.28		**0.14**	0.40
Average				0.35	0.33	0.56	0.61	**0.28**	0.87
Average				0.54	**0.53**		0.63		0.80

Table 3.14: The comparative HEO test results. Bolded items indicate the best of each group.

The proposed framework is easier to implement and has only one parameter to set (τ). The results obtained with HEO show that this is an effective and competitive approach. Taking into consideration the differences in processing times between machines where the experiments were performed (HEO in a 1.86 GHz Intel Core2

CPU and 917 MB of RAM versus VV-TS in a 3.2 GHz Intel P4 and 1 GB of RAM), the processing time for HEO is much less than the other methods. For example, the complete set of tests for the 270 problems solved using HEO required a computational time of around 34 hours. This contrasts with the maximum time required for VV-TS that solves only one problem (the most complex of them), in around 24 hours.

Thus, HEO was able to find near optimal values (less than one percent away) for the large test set of problems. Additionally, the results show that as the number of constraints increases, the percentage gap slowly rises. An important point that must be emphasised is that the value of τ used with the set of tests for small and large problems was different, $\tau = 1.4$ and $\tau = 2.2$ respectively. This indicates that the value of τ depends on the complexity of this problem (at the very least), a point that needs further investigation.

3.4.3 Large Test Data for the BPP

For the bin packing problem, the set of test problems contributed by Falkenauer [112] was used. This consists of 80 problems, which are divided into four groups of 20 problems. Each of these has 120, 250, 500 and 1000 items. The items' weights are spread uniformly between 20 and 100 and the capacity for all bins is 150. The name of the problem is defined with the nomenclature uNNN_PP, where NNN is the number of items and PP is the problem's identification. Each problem is accompanied by its theoretical optimal result.

3.4.3.1 Results

The number of generations for these problems is set to 100000 iterations because the local search used as the secondary search mechanism is more complex. The τ parameter was set at 1.4 as the best results were found with this value from the tested range $\{1.1 - 2.8\}$. The same τ value has been reported in previous research [143, 243, 42] as an appropriate value.

First, the extended local search mechanism for the BPP is tested to observe if the additional step applied to the local search method is an improvement. This part of the experiment describes the results when both methods are compared starting from the initial solution generated with the BFD strategy. Table 3.15 shows the results of the BSD method, the three step Falkenauer [112] method and the four step proposed method.

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U120_00	48	49	2.08	49	2.08	48	**0**
U120_01	49	49	**0**	49	**0**	49	**0**
U120_02	46	47	2.17	46	**0**	46	**0**
U120_03	49	50	2.04	50	2.04	49	**0**
U120_04	50	50	**0**	50	**0**	50	**0**
U120_05	48	49	2.08	48	**0**	48	**0**
U120_06	48	49	2.08	49	2.08	49	2.08
U120_07	49	50	2.04	50	2.04	49	**0**
U120_08	50	51	2	51	2	51	2
U120_09	46	47	2.17	47	2.17	47	2.17
U120_10	52	52	**0**	52	**0**	52	**0**
U120_11	49	50	2.04	50	2.04	49	**0**
U120_12	48	49	2.08	49	2.08	49	2.08
U120_13	49	49	**0**	49	**0**	49	**0**
U120_14	50	50	**0**	50	**0**	50	**0**
U120_15	48	49	2.08	49	2.08	48	**0**
U120_16	52	52	**0**	52	**0**	52	**0**
U120_17	52	53	1.92	53	1.92	53	1.92
U120_18	49	50	2.04	49	**0**	49	**0**
U120_19	49	50	2.04	50	2.04	50	2.04
Average			1.44		1.13		**0.62**

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U500_00	198	201	1.52	201	1.52	200	1.01
U500_01	201	204	1.49	203	1	202	0.5
U500_02	202	205	1.49	204	0.99	203	0.5
U500_03	204	207	1.47	207	1.47	206	0.98
U500_04	206	209	1.46	209	1.46	208	0.97
U500_05	206	207	0.49	207	0.49	206	**0**
U500_06	207	210	1.45	210	1.45	209	0.97
U500_07	204	207	1.47	207	1.47	206	0.98
U500_08	196	199	1.53	198	1.02	198	1.02
U500_09	202	204	0.99	203	0.5	202	**0**
U500_10	200	202	1	202	1	201	0.5
U500_11	200	203	1.5	203	1.5	202	1
U500_12	199	202	1.51	202	1.51	201	1.01
U500_13	196	198	1.02	198	1.02	196	**0**
U500_14	204	206	0.98	206	0.98	204	**0**
U500_15	201	204	1.49	203	1	202	0.5
U500_16	202	205	1.49	204	0.99	203	0.5
U500_17	198	201	1.52	201	1.52	200	1.01
U500_18	202	205	1.49	204	0.99	203	0.5
U500_19	196	199	1.53	198	1.02	199	1.53
Average			1.34		1.14		**0.67**

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U250_00	99	100	1.01	100	1.01	99	**0**
U250_01	100	101	1	101	1	100	**0**
U250_02	102	104	1.96	103	0.98	103	0.98
U250_03	100	101	1	101	1	100	**0**
U250_04	101	102	0.99	102	0.99	102	0.99
U250_05	101	104	2.97	103	1.98	102	0.99
U250_06	102	103	0.98	103	0.98	102	**0**
U250_07	103	105	1.94	104	0.97	104	0.97
U250_08	105	107	1.9	106	0.95	106	0.95
U250_09	101	102	0.99	102	0.99	102	0.99
U250_10	105	106	0.95	106	0.95	106	0.95
U250_11	101	103	1.98	102	0.99	102	0.99
U250_12	105	107	1.9	107	1.9	106	0.95
U250_13	102	104	1.96	104	1.96	103	0.98
U250_14	100	101	1	101	1	100	**0**
U250_15	105	107	1.9	107	1.9	106	0.95
U250_16	97	99	2.06	98	1.03	98	1.03
U250_17	100	101	1	101	1	100	**0**
U250_18	100	102	2	102	2	101	1
U250_19	102	103	0.98	103	0.98	102	**0**
Average			1.52		1.23		**0.64**

Problem name	Theo Opt	BFD	% gap	FLS	% gap	PLS	% gap
U1000_00	399	403	1	403	1	401	0.5
U1000_01	406	411	1.23	410	0.99	407	0.25
U1000_02	411	416	1.22	414	0.73	415	0.97
U1000_03	411	416	1.22	416	1.22	415	0.97
U1000_04	397	402	1.26	400	0.76	399	0.5
U1000_05	399	404	1.25	404	1.25	403	1
U1000_06	395	399	1.01	399	1.01	396	0.25
U1000_07	404	408	0.99	408	0.99	406	0.5
U1000_08	399	404	1.25	402	0.75	403	1
U1000_09	397	404	1.76	402	1.26	403	1.51
U1000_10	400	404	1	404	1	401	0.25
U1000_11	401	405	1	405	1	403	0.5
U1000_12	393	398	1.27	397	1.02	395	0.51
U1000_13	396	401	1.26	401	1.26	397	0.25
U1000_14	394	400	1.52	399	1.27	397	0.76
U1000_15	402	408	1.49	407	1.24	406	1
U1000_16	404	407	0.74	407	0.74	405	0.25
U1000_17	404	409	1.24	408	0.99	407	0.74
U1000_18	399	403	1	402	0.75	402	0.75
U1000_19	400	406	1.5	406	1.5	403	0.75
Average			1.21		1.04		**0.66**

Table 3.15: The results for the initial BSD solution, Falkenauer local search (FLS) and the proposed local search (PLS). Note that the number of bins and the percentage gap are shown. "Theo Opt" refers to the theoretical optimal number of bins.

Once the local search test was completed, the experiments with the HEO approach are performed and the results generated for this are compared with those reported by Martello and Toth (MTP) [203], Alvim, Glover and Ribeiro (LS-BPP) [11], Falkenauer (HGGA) [112], Levine and Ducatelle (HACO) [188], and Randall, Hendtlass and Lewis (SEO, PEO [3]) [244]. All the results are presented in the Tables 3.16, 3.17, 3.18 and 3.19. Each table represents a different group of problems.

[3]PEO means EO with a population of solutions.

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U120_00	48	**0**	**0**	**0**	**0**	**0**	**0**	2.08	**0**	**0**	**0**	**0**	**0**	**0**
U120_01	49	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_02	46	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_03	49	**0**	**0**	**0**	**0**	**0**	2.04	4.08	1.02	2.04	2.04	**0**	**0**	**0**
U120_04	50	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_05	48	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	2.08	**0**	**0**	**0**
U120_06	48	**0**	**0**	**0**	**0**	**0**	**0**	2.08	**0**	**0**	2.08	**0**	**0**	**0**
U120_07	49	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_08	50	2	2	2	**0**	**0**	**0**	1.96	**0**	**0**	**0**	**0**	**0**	**0**
U120_09	46	**0**	2.17	**0**	**0**	**0**	1.09	2.17	**0**	2.17	2.17	**0**	**0**	**0**
U120_10	52	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_11	49	**0**	**0**	**0**	**0**	**0**	**0**	2.04	**0**	**0**	**0**	**0**	**0**	**0**
U120_12	48	**0**	**0**	**0**	**0**	**0**	2.08	2.08	**0**	2.08	2.08	**0**	**0**	**0**
U120_13	49	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_14	50	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_15	48	**0**	**0**	**0**	**0**	**0**	**0**	2.08	**0**	**0**	**0**	**0**	**0**	**0**
U120_16	52	**0**	**0**	**0**	**0**	**0**	**0**	1.92	**0**	**0**	**0**	**0**	**0**	**0**
U120_17	52	**0**	**0**	**0**	**0**	**0**	**0**	5.77	**0**	1.92	1.92	**0**	**0**	**0**
U120_18	49	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U120_19	49	2.04	2.04	2.04	**0**	**0**	**0**	2	**0**	**0**	**0**	**0**	**0**	**0**
Average		0.2	0.31	0.2	**0**	**0**	0.26	1.41	0.05	0.41	0.62	**0**	**0**	**0**

Table 3.16: The comparative test results for the group of problems U120.

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U250_00	99	1.01	**0**	**0**	**0**	**0**	0.51	1.01	**0**	1.01	1.01	**0**	**0**	**0**
U250_01	100	**0**	**0**	**0**	**0**	**0**	**0**	1	**0**	**0**	1	**0**	**0**	**0**
U250_02	102	**0**	0.98	**0**	**0**	**0**	0.49	0.98	**0**	0.98	0.98	**0**	**0**	**0**
U250_03	100	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	1	**0**	**0**	**0**
U250_04	101	**0**	**0**	**0**	**0**	**0**	**0**	0.99	**0**	0.99	0.99	**0**	**0**	**0**
U250_05	101	1.98	0.99	**0**	**0**	**0**	0.99	1.98	0.99	0.99	1.98	**0**	**0**	0.99
U250_06	102	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
U250_07	103	0.97	0.97	0.97	**0**	**0**	**0**	3.85	**0**	**0**	**0**	0.97	0.97	0.97
U250_08	105	0.95	0.95	**0**	**0**	0.95	0.95	0.95	0.95	0.95	1.9	**0**	**0**	**0**
U250_09	101	0.99	**0**	**0**	**0**	**0**	0.99	2.97	0.99	0.99	0.99	**0**	**0**	**0**
U250_10	105	0.95	**0**	**0**	**0**	**0**	**0**	1.9	**0**	0.95	0.95	**0**	**0**	**0**
U250_11	101	0.99	0.99	**0**	**0**	0.99	0.99	0.99	0.99	0.99	0.99	**0**	**0**	**0**
U250_12	105	0.95	0.95	0.95	0.95	**0**	**0**	0.94	**0**	**0**	0.94	0.95	0.95	0.95
U250_13	102	0.98	0.98	0.98	0.98	**0**	**0**	0.97	**0**	**0**	0.97	0.98	0.98	0.98
U250_14	100	**0**	**0**	**0**	**0**	**0**	**0**	1	**0**	1	1	**0**	**0**	**0**
U250_15	105	0.95	0.95	**0**	**0**	0.95	0.95	3.81	0.95	1.9	1.9	**0**	0.95	0.95
U250_16	97	1.03	1.03	**0**	**0**	**0**	**0**	3.09	1.03	1.03	1.03	**0**	**0**	**0**
U250_17	100	**0**	**0**	**0**	**0**	**0**	**0**	1	**0**	**0**	**0**	**0**	**0**	**0**
U250_18	100	**0**	**0**	**0**	**0**	1	1	1	1	1	1	**0**	**0**	**0**
U250_19	102	**0**	**0**	**0**	**0**	**0**	**0**	0.98	**0**	**0**	0.98	**0**	**0**	**0**
Average		0.59	0.44	0.15	0.1	0.19	0.34	1.47	0.35	0.64	0.98	0.15	0.19	0.24

Table 3.17: The comparative test results for the group of problems U250.

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U500_00	198	1.52	0.51	**0**	**0**	0.51	0.51	2.53	0.51	1.01	1.01	**0**	**0**	**0**
U500_01	201	0.5	0.5	**0**	**0**	0.5	0.5	0.5	0.5	1	1	**0**	**0**	0.5
U500_02	202	0.99	**0**	**0**	**0**	**0**	0.5	0.99	0.5	0.74	0.99	**0**	**0**	**0**
U500_03	204	0.98	0.49	**0**	**0**	0.49	0.49	1.47	0.98	0.98	0.98	**0**	**0**	0.49
U500_04	206	1.46	**0**	**0**	**0**	**0**	0.49	1.94	0.49	0.49	0.97	**0**	**0**	**0**
U500_05	206	0.49	**0**	**0**	**0**	**0**	0.97	2.91	0.49	0.73	0.97	**0**	**0**	**0**
U500_06	207	1.45	0.48	**0**	**0**	0.48	0.72	1.45	0.97	0.97	1.45	**0**	**0**	0.48
U500_07	204	1.47	0.49	**0**	**0**	0.49	1.23	3.43	0.98	1.47	1.96	**0**	0.25	0.49
U500_08	196	1.02	0.51	2	**0**	**0**	0.26	1.02	0.51	0.51	1.02	**0**	**0**	0.51
U500_09	202	0.99	**0**	**0**	**0**	**0**	**0**	0.5	**0**	0.5	0.99	**0**	**0**	**0**
U500_10	200	1	0.5	**0**	**0**	**0**	0.5	0.5	0.5	0.5	1	**0**	**0**	**0**
U500_11	200	1	0.5	**0**	**0**	0.5	0.5	3	0.5	1	1.5	**0**	**0**	0.5
U500_12	199	1.51	0.5	**0**	**0**	0.5	0.5	1.01	0.5	1.01	1.01	**0**	**0**	0.5
U500_13	196	0.51	**0**	**0**	**0**	**0**	0.51	1.53	**0**	0.51	0.51	**0**	**0**	**0**
U500_14	204	0.49	**0**	**0**	**0**	0.49	0.49	8.33	0.49	0.49	0.98	**0**	**0**	**0**
U500_15	201	1	**0**	**0**	**0**	0.5	0.5	0.5	0.5	0.5	1	**0**	**0**	**0**
U500_16	202	0.99	**0**	**0**	**0**	**0**	**0**	0.99	**0**	0.5	0.5	**0**	**0**	**0**
U500_17	198	1.52	**0**	**0**	**0**	0.51	0.51	0.51	0.51	1.01	1.01	**0**	**0**	0.51
U500_18	202	1.49	**0**	**0**	**0**	**0**	0.5	1.98	0.99	1.49	1.49	**0**	**0**	**0**
U500_19	196	1.53	0.51	2.04	**0**	0.51	0.51	1.02	0.51	1.02	1.53	**0**	**0**	**0**
Average		1.09	0.25	0.2	**0**	0.27	0.51	1.81	0.52	0.82	1.09	**0**	0.01	0.02

Table 3.18: The comparative test results for the group of problems U500.

Problem	Theo	MTP	LS-BPP	HGGA	HACO	SEO			PEO			HEO		
name	Opt	%	%	%	%	mi%	me%	ma%	mi%	me%	ma%	mi%	me%	ma%
U1000_00	399	1	**0**	**0**	**0**							**0**	**0**	0.25
U1000_01	406	0.99	**0**	**0**	**0**							**0**	**0**	0.25
U1000_02	411	1.22	**0**	**0**	**0**							**0**	**0**	**0**
U1000_03	411	1.22	0.49	**0**	**0**							**0**	0.24	0.49
U1000_04	397	1.01	0.25	**0**	**0**							**0**	0.25	0.5
U1000_05	399	0.75	0.25	**0**	**0**							**0**	**0**	0.25
U1000_06	395	0.76	**0**	**0**	**0**							**0**	**0**	**0**
U1000_07	404	0.5	**0**	**0**	**0**							**0**	**0**	0.25
U1000_08	399	0.75	**0**	**0**	**0**							**0**	**0**	0.25
U1000_09	397	1.26	0.5	**0**	**0**							**0**	**0**	0.25
U1000_10	400	1	**0**	**0**	**0**							**0**	**0**	**0**
U1000_11	401	0.75	0.25	**0**	**0**							**0**	**0**	0.25
U1000_12	393	0.76	**0**	**0**	**0**							**0**	**0**	0.25
U1000_13	396	1.26	**0**	**0**	**0**							**0**	0.25	0.25
U1000_14	394	1.27	0.51	**0**	**0**							**0**	0.13	0.25
U1000_15	402	1.24	0.25	**0**	**0**							**0**	**0**	0.25
U1000_16	404	0.74	**0**	**0**	**0**							**0**	**0**	0.25
U1000_17	404	0.74	0.25	**0**	**0**							**0**	**0**	0.25
U1000_18	399	1	**0**	**0**	**0**							**0**	**0**	**0**
U1000_19	400	1.25	**0**	**0**	**0**							**0**	**0**	0.25
Average		0.97	0.14	**0**	**0**							**0**	0.04	0.22

Table 3.19: The comparative test results for the group of problems U1000.

For the four methods that do not use extremal optimisation, the results are presented only using one column that shows the percentage gap with respect to the theoretical optimal value. For the remaining three methods that use extremal optimisation, the results are presented by three columns, "mi%", "me%", and "ma%", which denote the minimum, median and maximum percentage gap respectively. Note that blank spaces in Table 3.19 are present because the authors did not solve these problems. In Table 3.20 the HEO's runtimes in relation to the other methods are shown.

Approach \ Problem group		U120_Group	U250_Group	U500_Group	U1000_Group
MTP	average time	370.00	1516.00	1535.00	9393.00
LS-BPP	average time	**0.20**	6.70	37.25	143.55
HGGA	average time	381.00	1337.00	1015.00	7059.00
HACO	average time	1.00	52.00	50.00	147.00
HEO	average time	1.00	**1.20**	**1.20**	**1.80**

Table 3.20: The average computational time, in seconds. Bolded items indicate the shortest time within the groups.

3.4.3.2 Discussion

As can be seen in Table 3.15, the improvement made in the new local search version with respect to Falkenauer's method gave better results for 52 out of 74 problems where the initial solution is not optimal. Additionally, the average percentage gap for the four groups decreased from around 1.1% to under 0.67%. This means that for the four groups, the improved local search method was able to find closer results to the theoretical solution. Thus, the improved local search method is a good choice to be used as a secondary search mechanism in the HEO framework to solve the BPP.

Observing the data from Tables 3.16 to 3.19, HEO can be seen as an efficient and competitive approach to solve the BPP. In 77 out of 80 problems, the theoretical optimal result was found, only one less than HACO [188], which found 78. The result of the three remaining problems (u250_07, u250_12, u250_13) is only one bin from the theoretical optimal solution. In fact, the problem u250_13 has no solution for the theoretical optimal value, as reported in Levine and Ducatelle [188].

Despite the good results obtained by the previous two works, where extremal optimisation was applied (SEO and PEO), it can be conjectured from the development of HEO that the better results are due to the use of a good initial solution, the

improved local search mechanism and the modified extremal optimisation selection process.

As has been mentioned previously, the extremal optimisation meta-heuristic is a simple and computationally inexpensive algorithm. This feature also can be seen reflected in Table 3.18 where the average time necessary to run the algorithm is less than the other methods (with the exception of the first group of 120 items where LS-BPP had a better time) and the difference in the average time among the four test groups is quite similar, which shows a balanced performance in relation to the number of items to be packed.

3.4.4 Large Test Data for the GAP

A set of twenty-four large-sized problems proposed by Chu and Beasley [62] taken from the OR-Library [24] were solved. These problems are classified in four groups (A, B, C, D) according to their level of hardness of the constraints. The following definitions are reproduced from Chu and Beasley [62].

Group A: a_{ij} are integers from $U(5,25)$, c_{ij} are integers from $U(10,50)$ and $b_j = 0.6\left(\frac{n}{m}\right)15+0.4R$ where $R = \max_{j\in J}\sum_{i\in I,J_i=j} a_{ij}$ and $J_i = \min[j|c_{ij} \leqslant c_{ik}, \forall k \in J]$.

Group B: a_{ij} and c_{ij} are the same as Group A and b_j is set to 70% of the value given in Group A.

Group C: a_{ij} and c_{ij} are the same as Group A and $b_j = 0.8\sum_{i\in I}\frac{a_{ij}}{m}$.

Group D: a_{ij} are integers from $U(1,100)$, $c_{ij} = 111 - a_{ij} + e$ where e are integers from $U(-10,10)$ and $b_j = 0.8\sum_{i\in I}\frac{a_{ij}}{m}$.

Each group is composed of six problems that can have a number of agents $m \in \{5,10,20\}$ and a number of jobs $n \in \{100,200\}$.

The names of the problems are defined by the nomenclature $G.M.N$ where G is the name of the group, M is the number of agents and N is the number of jobs. Table 3.21 shows the groups A, B and C that are accompanied by the optimal solutions for each problem and the group D that is accompanied by the theoretical solution found by the linear programming method [113].

Name Problem	Optimal Solution	Name Problem	Optimal Solution	Name Problem	Optimal Solution	Name Problem	Theoretical Best Solution
A.5.100	1698	B.5.100	1843	C.5.100	1931	D.5.100	6353
A.5.200	3235	B.5.200	3552	C.5.200	3456	D.5.200	12736.2
A.10.100	1360	B.10.100	1407	C.10.100	1402	D.10.100	6323.4
A.10.200	2623	B.10.200	2827	C.10.200	2806	D.10.200	12418.3
A.20.100	1158	B.20.100	1166	C.20.100	1243	D.20.100	6142.5
A.10.200	2339	B.20.200	2339	C.20.200	2391	D.20.200	12217.7

Table 3.21: The benchmark case tests for the generalised assignment problem.

3.4.4.1 Results

For the generalised assignment problem, the number of iterations was set to 500000, as in the multi-dimensional knapsack problem. The assignment of the value for the τ parameter was carried out through a single run of 100000 iterations for each instance, testing a range of values from 1.1 to 2.9 in increments of 0.1. Just like in the large test data for the multi-dimensional knapsack problem, the best results were achieved with a τ value equal to 2.2.

The results generated by the HEO approach were compared with those obtained by Chu and Beasley (CB-GA) [60] using genetic algorithms; Lorena, Narciso and Beasley (CGA) [196] with a constructive genetic algorithm, Feltl and Raidl (CRHGA) [113] through a constraint-ration heuristic applied with genetic algorithms; and Randall, Hendtlass and Lewis (SEO, PEO) [243, 244] using a single and population version of extremal optimisation.

Table 3.22 shows the best and median percentage gap for each problem with respect to the theoretical best values that are illustrated in Table 3.21. However, the median percentage gap data for the problem CGA were not available and for the problem CRHGA only the information about the average percentage gap for each instance was available. The latter was incorporated as a reference to obtain a more complete picture about the result achieved by the CRHGA approach.

3.4.4.2 Discussion

For group A that represents the less constrained and easy set of problems to solve, HEO was always capable of discovering every optimal solution. The same results were reported for the other methods.

Name Problem	LP Optimal Value	CB-GA best %gap	CB-GA med %gap	CGA best %gap	CGA med %gap	CRH-GA best %gap	CRH-GA ave %gap	SEO best %gap	SEO med %gap	PEO best %gap	PEO med %gap	HEO best %gap	HEO med %gap
A.5.100	1698	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
A.5.200	3235	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
A.10.100	1360	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
A.10.200	2623	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
A.20.100	1158	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
A.10.200	2339	**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
A Group Average		**0**	**0**	**0**		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
B.5.100	1843	**0**	0.24	**0**		**0**	**0.03**	0.71	1.03	**0**	**0.03**	0.11	0.22
B.5.200	3552	**0.03**	0.34	1.38		**0.03**	0.12	0.45	0.51	0.08	**0.10**	0.08	0.20
B.10.100	1407	**0**	0.07	0.21		**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
B.10.200	2827	0.14	0.32	0.14		**0.07**	0.26	0.74	0.90	0.14	0.25	**0.07**	**0.21**
B.20.100	1166	**0**	0.09	**0**		**0**	0.14	0.09	0.17	**0**	**0**	**0**	0.13
B.20.200	2339	**0.04**	0.09	0.34		**0.04**	**0.12**	0.21	0.26	**0.04**	0.15	0.09	0.17
B Group Average		0.04	0.19	0.35		**0.02**	0.11	0.37	0.48	0.04	**0.09**	0.06	0.15
C.5.100	1931	**0**	0.44	0.52		**0**	0.15	0.36	0.60	**0**	**0.05**	**0**	0.21
C.5.200	3456	0.06	0.23	0.12		0.03	**0.07**	0.35	0.58	0.06	0.10	**0**	0.16
C.10.100	1402	**0.07**	0.14	1.50		**0.07**	**0.11**	0.78	1.18	0.14	0.21	**0.07**	0.21
C.10.200	2806	0.29	0.52	0.32		**0.04**	**0.22**	0.75	1.10	0.29	0.29	0.18	0.32
C.20.100	1243	0.08	0.48	0.08		**0**	0.26	0.80	1.13	0.08	**0.16**	0.16	0.32
C.20.200	2391	0.25	0.67	0.25		**0.21**	0.56	1.17	1.42	0.25	**0.25**	0.29	0.54
C Group Average		0.12	0.41	0.46		**0.06**	0.23	0.70	1.00	0.14	**0.18**	0.12	0.29
D.5.100	6353	0.31	0.65	1.98		**0.19**	**0.31**	1.68	1.86	0.57	0.76	0.27	0.42
D.5.200	12736.2	0.47	0.70	0.68		**0.24**	**0.39**	1.99	2.10	0.39	0.74	0.33	0.43
D.10.100	6323.4	0.88	1.70	1.05		**0.78**	**1.24**	3.05	3.46	1.37	1.62	1.35	1.61
D.10.200	12418.3	1.47	1.72	1.74		**0.95**	1.17	2.73	3.03	1.47	1.51	**0.95**	**1.08**
D.20.100	6142.5	2.06	2.74	2.24		**1.90**	**2.36**	2.40	4.58	2.40	2.52	2.30	2.62
D.20.200	12217.7	1.92	2.58	2.07		**1.38**	2.00	1.92	3.64	1.92	2.09	1.75	**1.84**
D Group Average		1.19	1.68	1.63		**0.91**	**1.25**	2.86	3.11	1.39	1.54	1.16	1.34
Test Set Average		0.34	0.57	0.61		**0.25**	**0.40**	0.98	1.15	0.39	0.45	0.33	0.44

Table 3.22: The results of the benchmark case tests for the generalised assignment problem. Bolded items indicate the best for each instance.

For groups B and C that represent the problems with an intermediate level of constraint difficulty, HEO was a competitive approach that was able to find some optimal results having a similar performance compared to CB-GA and PEO. Although the average percentage gap of HEO for groups B and C were not better than CB-GA and PEO, HEO found similar or superior results for 7 out of 12 problems, of which, problems B.10.200, C.5.200 and C.10.200, HEO outperformed the other two methods. The CRHGA approach had the best results for most problems with the exception of problem C.5.200, where HEO was able to find the optimal value. This result can be interpreted that HEO is an effective approach able to explore remote zones in the search space.

For group D that represents the problems with the hardest level of difficulty on the constraints, HEO presented an excellent performance only surpassed by CRHGA. In fact, HEO was the only method that could find a similar value for one of the problems in group D (D.10.200) when compared to CRHGA.

The effectiveness of CRHGA is based on a complex hybrid algorithm, which in turn is an improvement of the CB-GA approach. CRHGA uses constraint-ratio and linear programming heuristics to generate the initial solution. Furthermore, the selection and replacement mechanism was modified using a relative fitness assignment, as also the mutation operator was adapted to use more elaborate rules to change the solution. Both the CB-GA and the CRHGA techniques require a large number of iterations to find the optimal or approximate optimal solution. Even though CRHGA performs much better than its predecessor (CB-GA), it was necessary to have more than 1000000 iterations, even reaching around 2500000 iterations for the most constrained problems in group D as reported.

When HEO was compared with other implementations of EO, the following points can be made. The reasons why HEO outperforms SEO could be in the implementation of the differentiated fitness evaluation scheme and the double traverse local search applied to improve the feasible solutions found by the extremal optimisation mechanism. These extensions even enabled HEO to achieve a similar performance to PEO, which is a population-based version of SEO. This latter fact leads to the belief that the extension of HEO to work with a population of solutions could allow it to reach better results. This is an issue that must be strongly considered in future work.

Note that the single parameter τ used in HEO was set with the value 2.2, the same value used with the large test data for the multi-dimensional knapsack problem. This value differs from the commonly used value of 1.4 reported by Boettcher and Percus [39, 41, 44], Randall [243], Randall and Lewis [245] and Randall, Hendtlass and Lewis [244]. This is another aspect that requires more investigation in future work.

3.5 Summary

This chapter described the HEO framework to solve constrained combinatorial optimisation problems. The strength of HEO lies in the joint work of the constraint handling scheme with the secondary search mechanism incorporated into the canonical extremal optimisation meta-heuristic. The proposed differentiated fitness evaluation scheme is applied to deal with the feasible or infeasible solutions generated by the extremal optimisation mechanism, which in turn is used as the main framework to carry out a coarse-grain exploration of the search space. This first extension to extremal optimisation allows it to direct the search through feasible and infeasible regions looking for the optimal solution. The secondary search mechanism is used as a complementary

technique to carry out a fine-grain search on the search space by refining each new feasible solution found in the main framework. The HEO approach maintains the advantage of the canonical extremal optimisation meta-heuristic in regard to its minimal requirement for parameterisation and implementation compared to other, more complex, methods.

The differentiated fitness evaluation assesses the level of optimality or violation of the constraints for each component of the solution. The secondary search mechanism is implemented by a double traverse local search that performs a swap of components in the solution representation. Within the context of the test problems, HEO is able to be applied to an exploratory number of problems and the computational study performed in some of them has shown that these two proposals, working together, are an effective and efficient way to obtain competitive results.

For the MKP, the proposal shows optimal results for the small test problems and competitive results with the more intricate large test problems. For the BPP, the results obtained were very good, reaching optimal results as well as those achieved by the most successful method so far. For the GAP, HEO was able to deliver very good quality solutions and therefore is competitive with more elaborate methods.

The results presented herein are promising. Future work will include solving more complex test problems for the MKP, BPP and GAP, as well as solving the constrained combinatorial optimisation problems that were described above, such as the frequency assignment problem, the capacitated vehicle routing problems, the single source capacitated facility location problem, the capacitated minimal spanning tree problem and the capacitated set covering problem. An interesting point to be developed is to perform a more detailed analysis regarding the effects of local search in the proposed HEO framework through the comparison of the techniques used in this chapter without local search. This will determine the level of coarse grain search power that the technique possesses in comparison to other nature-inspired meta-heuristics. Additionally, the idea of adding a population-based extension to HEO, with the objective of improving the results, is a promising avenue of future research.

Finally, the τ value of 1.4 was corroborated as a good value to obtain efficient solutions for the small test data for the MKP (Subsection 3.4.1) and the large test data for the BPP (Subsection 3.4.3), as has been reported in previous research [42, 143, 243]. However, the τ value 2.2 used with the large test data for the MKP (Subsection 3.4.2) and the large test data for the GAP (Subsection 3.4.4) would suggest that there could be more than a single good value to obtain efficient solutions, and this could be potentially sensitive to problem complexity. An interesting challenge could be to work

with different τ values for different types of problems, and thereby find the τ value that is able to achieve the best results for each of them.

3.6 Contributions

Based on the theoretical development and the results of the experimentation phase, it is evident that the following two objectives have been achieved:

- To incorporate a constraint-handling mechanism that allows extremal optimisation to deal with infeasible solutions.
- To provide a hybrid extremal optimisation framework to solve single-objective constrained combinatorial optimisation problems.

With the fulfilment of these objectives, it is believed that the work developed in this chapter has contributed to the knowledge of constraint handling, extremal optimisation and hybrid methods for CCOPs in the following ways:

- The differentiated fitness evaluation scheme is proposed as a novel constraint handling technique for extremal optimisation. This scheme allows extremal optimisation to solve not only problems where the representation of the solution generates exclusively feasible solutions, but also those that generate infeasible solutions.
- A component fitness evaluation based on the level of optimality or violation of constraints is established to complement the differentiated fitness evaluation scheme. This provides a formal criterion to evaluate each component of the solution instead of evaluating the complete solution (a mechanism widely used in evolutionary algorithms).
- A generalisable secondary search mechanism based on memetic algorithms is incorporated into the extremal optimisation meta-heuristic as a method to improve the convergence of solutions. This diversifies the way that the search is performed in the neighbourhood of the last feasible solution that was found, which improved the convergence of HEO.
- The double traverse local search algorithm is proposed as a general secondary search mechanism. This algorithm develops a fine-grained search in the neighbourhood of the last feasible solution found, taking advantage of the representa-

tion of the solution used in HEO. Thus, it is the first proposal of several future mechanisms that will be incorporated as the secondary search mechanism.

- A hybrid extremal optimisation framework is generated as an improvement to the extremal optimisation meta-heuristic. This allows the exploration capacity of HEO to be enhanced.

- An exploratory and novel hybrid method based on extremal optimisation is incorporated into the nature-inspired algorithm to offer the possibility of solving capacitated constrained combinatorial optimisation problems. This permits the nature-inspired search scheme, used by extremal optimisation, to be applied to a new range of combinatorial optimisation problems. Also, this gives the opportunity for new CCOPs be solved by a novel meta-heuristic, like extremal optimisation, with the possibility of finding more efficient and effective results.

- A number of derivative research areas emerge from the work developed in this chapter, as mentioned in Section 3.5. This offers the possibility to investigate the current exploration capacity and efficiency of HEO, as also the addition of new features that enhance the HEO mechanism to solve different types of problems.

Chapter 4

A Multi-Objective Hybrid Extremal Optimisation Framework

4.1 Introduction

Extremal optimisation is a relatively recent nature-inspired meta-heuristic as it has been described in previous chapters. In this meta-heurisitc, the search method is especially suitable for solving combinatorial optimisation problems classified as $\mathcal{NP}$-hard, particularly those having multiple feasible and infeasible regions (see Figure 3.2). Extremal optimisation has the feature of requiring only one algorithm-specific parameter and it uses a minimal amount of memory, which can often lead to a lower computation time. Hence, this novel method is giving a new perspective on solving complex combinatorial problems rather than using the traditional evolutionary algorithms as was discussed in Section 2.3.2.1.

In the previous chapter, an extension of the canonical extremal optimisation to handle generic restrictions, and improve the solution convergence through the hybrid extremal optimisation (HEO) framework has been proposed. This enhancement allows extremal optimisation to solve constrained combinatorial problems, especially those with capacity restrictions, which expands the range of optimisation problems that can be addressed.

However, to date most research in extremal optimisation has been applied to solve single-objective problems and there have only been a relatively small number of attempts to extend it towards multi-objective problems (see Section 2.3.2.2). This fact

has motivated the development in this chapter of a multi-objective version of the HEO framework (MOHEO) to solve multi-objective combinatorial problems.

A *new multi-objective hybrid extremal optimisation* framework proposal is presented here. This develops an extension of both the differentiated fitness evaluation scheme, to calculate the fitness for two or more objective functions, and the secondary search mechanism represented by the double traverse local search, to improve the exploration according to the different objective functions. The former allows extremal optimisation to perform a coarse-grain search on the landscape, enabling the framework to find new non-dominated points, as the solution moves closer to the Pareto-front. The latter allows a fine-grain search to obtain a better approximation of the Pareto-front as well as to produce a more diverse set of points near the ends of the Pareto-front. The collaborative uses of both mechanisms, which are incorporated into the multi-objective hybrid extremal optimisation framework, are suitable (in principle) to solve a range of multi-objective combinatorial optimisation problems. The advantage of this proposal is the simplicity of its implementation, with only two parameters to be set, and the efficiency of its performance.

Given this new framework, the MOHEO operation is tested through a set of benchmark problem instances for multi-objective combinatorial optimisation. All the problem instances used in this chapter include their experimental results, which are used as reference solutions. Thus, the multi-objective 0/1 knapsack problem (MOKP) with the benchmark instances taken from the research work developed by Zitzler and Laumanns [320, 324, 322], the multi-objective quadratic assignment problem (MOQAP) with the benchmark instances taken from the research work developed by Knowles and Corne [165, 167, 168] and the multi-objective permutation flow-shop scheduling problem (MOFSSP) with the benchmark instances taken from the research work developed by Ishibuchi, Yoshida and Murata [154, 152], were selected.

Results show that the new framework is able to obtain competitive solutions when these are compared with those obtained by traditional methods such as SPEA2 and NSGA-II. The non-dominated points found are well-distributed and similar, or very close, to the approximated Pareto-front set found by the benchmark instances. Furthermore, an interesting point to be considered is the difficulty in finding a proper fitness assessment for each component of the solution. This issue was highlighted with the multi-objective permutation flow-shop scheduling problem. Thus, taking into consideration the promising results achieved by MOHEO, it is believed that the multi-objective extension of HEO can be a potential choice to efficiently solve multi-objective combinatorial optimisation problems.

The objective of this chapter is to provide a hybrid extremal optimisation framework to solve multi-objective constrained combinatorial optimisation problems. The rest of this chapter is organised as follows. Section 4.2 explains the multi-objective hybrid extremal optimisation framework to solve multi-objective combinatorial optimisation problems. Section 4.3 presents how the multi-objective hybrid extremal optimisation is applied to the selected multi-objective combinatorial optimisation problems. Section 4.4 shows a summary of the computational experiments developed with an analysis of them. Section 4.5 gives a summary of the chapter and discusses the future work arising from this study. Finally, Section 4.6 states the contributions that emerge from this chapter.

4.2 MOHEO for Multi-Objective CCOPs

This section describes the multi-objective hybrid extremal optimisation framework. First, the extension of the differentiated fitness evaluation scheme for multi-objective optimisation is defined. Then, the secondary search mechanism is adapted to search in a multi-objective search space, as a complement to extend the exploration of new points of the Pareto-front set. Finally, the integrated framework to solve multi-objective combinatorial optimisation problems is given.

4.2.1 Multi-Objective Fitness Evaluation Scheme

According to Fonseca and Fleming [118], the fitness evaluation functions used in multi-objective optimisation evolutionary algorithms can be categorised into three groups. These groups are: the Pareto-based approaches, the non-Pareto-based approaches and the aggregating function approaches.

Pareto-based approaches are population-based mechanisms that carry out a ranking for each individual in the population based on the Pareto-dominance relation ($\prec$) in order to establish the probability of selection for the procreation process. The main idea is to identify the non-dominated individuals in the population, which have special treatment to propagate their genetic information in future generations in order to improve the species according to the different objective functions. Some Pareto-based approaches are NSGA-II [100], SPEA2 [322] and NPGA2 [111].

Non-Pareto-based approaches are also population-based techniques that are based on other principles to perform the fitness evaluation such as the special manipulation of the objectives, particular selection criteria or special population rules. As a non-Pareto-based mechanism, one additional step is necessary and must be applied on each

generation's offspring to discover new non-dominated solutions. The most known non-Pareto-based approach is VEGA [254, 255, 256], which performs a modified selection procedure. Here, the population with M individuals is divided into m sub-populations of M/m individuals, where m is the number of objectives. Each sub-population is filled with a proportional selection of individuals according to each objective function; that is, each of them evolves towards a single objective. Then the individuals of each sub-population are shuffled to generate a population of size M on which the traditional crossover and mutation operator are applied.

Aggregating functions are mainly used by single-individual methods. These approaches join independent evaluations, one for each objective, in a single value that is assigned as the fitness for the individual. This technique generally requires a deep knowledge of the problem domain and, in contrast to the population-based approaches, it also demands an extra iteration process to generate the approximated Pareto-front set. The most popular aggregating function approach is the weighted-sum method [118, 128, 280, 313].

Of these three approaches, the one that best fits with the hybrid extremal optimisation framework is the aggregating function technique. This is due firstly to the single solution scheme that is performed by extremal optimisation and secondly because it is a reasonable extension for the selection mechanism that must carry out a component fitness evaluation from the solution representation. However, the implementation of a Pareto-based component fitness evaluation is an interesting challenge to be taken up in future research work.

The differentiated fitness evaluation scheme represented by Figure 3.3 and Equation 3.2 is taken as the basis to implement the multi-objective differentiated fitness evaluation version. Recall that when the solution is feasible, the fitness assessment is focussed on locating which part of the solution adversely affects its quality the most. However, when the solution is infeasible, the fitness calculation is focussed on locating which part of the solution is more responsible for the level of violation of one or more constraints. Thus, the extension from the single-objective hybrid extremal optimisation fitness evaluation to the multi-objective version is given by the simple scalar addition of the fitness for each objective function presented in the problem to be solved. This aggregating function is shown in Equation 4.1.

$$\Lambda(x_i) = \sum_{j=1}^{m} \lambda^k(x_i), \quad \forall i \quad 1 \leq i \leq n \tag{4.1}$$

where:

m	is the number of objectives,
n	is the number of components in the solution,
$\lambda^k(x_i)$	is the k^{th} single-objective fitness function applied to the i^{th} component using as its base Equation 3.3, and
$\Lambda(x_i)$	is the multi-objective fitness of the i^{th} component.

Knowing that the proposed aggregation function will carry out an exploration biassed toward the centre part of the Pareto-front surface, as is shown in Figure 4.1, the secondary search mechanism will be responsible for the exploration of the ends of the surface. A metaphor of this may be the nomadic pastoralism route where an individual shepherd permanently moves on a landscape looking for better places to find fresh pasture for his/her livestock. In this case, the search is focussed on finding non-dominated points.

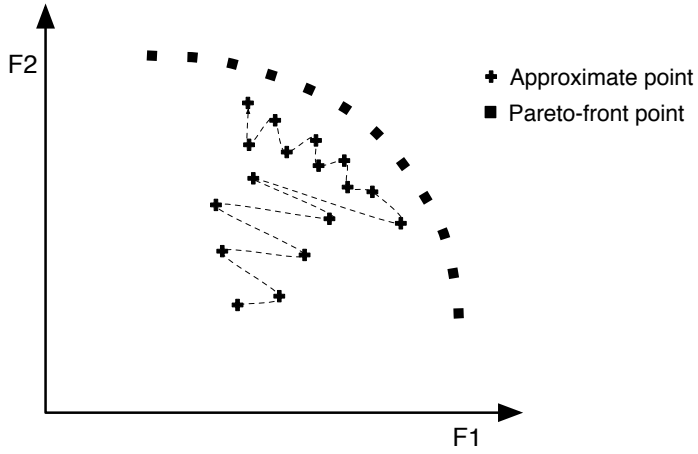


Figure 4.1: The aggregating function approximation to the Pareto-front for a bi-objective maximisation problem. F1 and F2 represent the two objective functions.

4.2.2 A Multi-Objective Secondary Search Mechanism

As in single-objective optimisation, there have been an increasing number of multi-objective approaches that have incorporated local search methods to complement the main search mechanism with the objective of improving results. To accomplish this, the two approaches often developed are based on a Pareto-ranking and weighted scalar fitness functions [153]. Comparative studies, showing the advantages and disadvantages of both techniques, can be found in Ishibuchi and Narukawa [153] and Jaszkiewicz [156].

The aim of this subsection is to propose an extension to the secondary search

mechanism based on the double traverse local search technique that was presented in Algorithm 7. This enhancement has a dual purpose. First, it tries to improve the convergence for the last feasible point that has been found at the current iteration to find new non-dominated points towards the true Pareto-front. Secondly, new points near the ends of the approximated surface, by means of a random nomadic pastoralism route on the search space targeted towards the true Pareto-front, are discovered.

After having studied and analysed the two local search techniques mostly used in multi-objective evolutionary algorithms, a novel and simple idea based on a modified weighted scalar function emerges to implement the multi-objective secondary search mechanism, using the double traverse local search presented for single-objective optimisation as the basis.

The multi-objective version of the double traverse local search mechanism is not unduly complicated from its original form. The proposal is based on the modification and mixture of the concepts related to the weighted scalar fitness and the lexicographic ordering.

Firstly, in the implementation of the weighted scalar fitness, the weighted vector is replaced by a probability vector, which is filled with random values $P(m)_{rand} \sim U(1,0)$. This probability vector is used to choose an objective, based on its assigned probability P_i, using the roulette wheel selection technique. The chosen objective is then used to evaluate and guide the local search through a cycle. After that, a new objective function is selected to carry out a new search in another direction, which could be the same as before.

As can be seen, the main feature of the proposed multi-objective local search lies in its ability to change the orientation of the search according to the objective function chosen to evaluate the quality of the new points or solutions that are being discovered along the path. The multi-objective double traverse local search performs as follows:

- First, the information associated with one component indexed by the first traverse is interchanged by the information associated with one component indexed by the second traverse.

- If, as result of this interchange, a better solution is produced according to the current objective function selected with probability P_i, then the interchange becomes effective, otherwise the components keep their values unchanged.

- The same analysis is performed below when the interchange is carried out among one component indexed by the first traverse and by two components indexed by the second traverse.

The interesting part of this method is that, in the multi-objective version before each single traverse is started, a new objective function is selected, which could be the same as before. This selection of the objective gives to the current solution the possibility to explore new places in the search space without being influenced entirely by a particular objective. Thus, different objective functions are used according to its individual probability P_i along the local search process. Each time that the secondary search mechanism is called, a new probability vector is generated. This allows all objective functions to have the opportunity to lead to any double traverse local search process. Algorithm 15 shows this novel proposed approach, which is an $O(n^2)$ algorithm where n is the number of components of the solution.

Algorithm 15 The general multi-objective double traverse local search pseudocode.

```
1: Generate a random variate vector P(m)_rand ~ U(1,0)
2: repeat
3:   {First Step}
4:   for each component i in the solution structure do
5:     Select an objective using RWS and the P(m)_rand probability
6:     for each component j different from i in the solution structure do
7:       if the value swap between components i and j is possible AND the solution
         remains feasible AND as a result of this operation is generated a better
         solution according to the selected objective then
8:         Implement the operation and update the variables involved
9:         if the new solution is non-dominated then
10:          Incorporate it into the approximated Pareto-front set found so far
11:        end if
12:      end if
13:    end for
14:  end for
15:  {Second Step}
16:  for each component i in the solution structure do
17:    Select an objective using RWS and the P(m)_rand probability
18:    for each components j and j' different from i in the solution structure do
19:      if the value swap between component i and components j and j' is possible
         AND the solution remains feasible AND as a result of this operation is
         generated a better solution according to the selected objective then
20:        Implement the operation and update the variables involved
21:        if the new solution is non-dominated then
22:          Incorporate it into the approximated Pareto-front set found so far
23:        end if
24:      end if
25:    end for
26:  end for
27: until there is no improvement in the solution
```

Finally, if the local search finds a better solution then the non-dominance procedure is called to see if the new solution should be added to the approximated Pareto-front set found up to now. Thus, Figure 4.2 illustrates how, from the last feasible solution that was found, the double traverse local search mechanism starts a route looking for new non-dominated solutions.

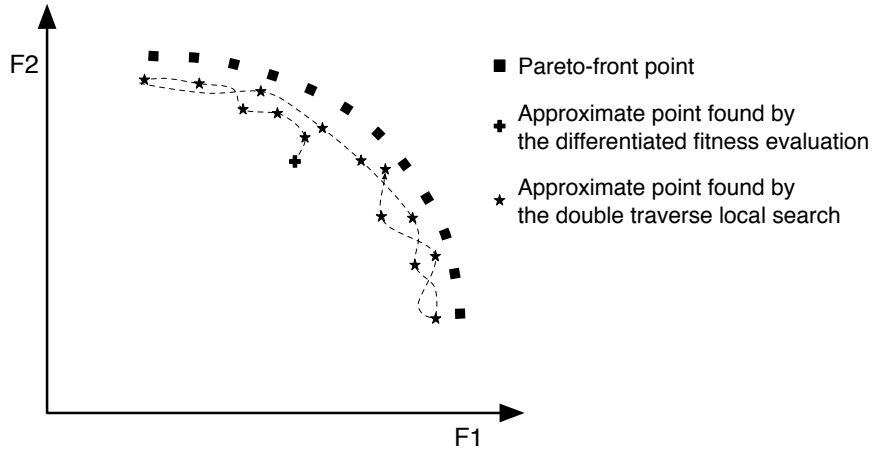


Figure 4.2: The double traverse local search approximation to the Pareto-front for a bi-objective maximisation problem. F1 and F2 represent the two objective functions.

4.2.3 The Multi-Objective Hybrid Extremal Optimisation Framework

The multi-objective hybrid extremal optimisation framework proposed in this chapter is based on the single-objective framework described in Chapter 3. Herein, an initial multi-objective framework that expands the scope of extremal optimisation toward multi-objective problems, is presented. This framework is implemented through the enhancement of both the differentiated fitness evaluation scheme and the secondary search mechanism to handle multiple objectives.

A relevant consideration that must be taken into account is the fact that in multi-objective optimisation there is not only one best solution, which has to be improved along the evolutionary process, but an indeterminate amount of solutions that make up the approximated Pareto-front set. For this reason, MOHEO works by simulating the random nomadic pastoralism route [110] of nomadic peoples looking for the best places to live temporarily. The different points that belong to the true Pareto-front set represent those best places. Thus, the single-solution scheme, which works with extremal optimisation, is akin to one of these communities' trips and each time that a good place (non-dominated Pareto-point) is found, this is registered in the community

records (approximated Pareto-front set).

Figure 4.3 presents this novel general multi-objective hybrid extremal optimisation framework. The modules that have not changed with respect to the description given in Section 3.2.3 will not be discussed again. Only the new modules and the modules that have been modified will be explained below.

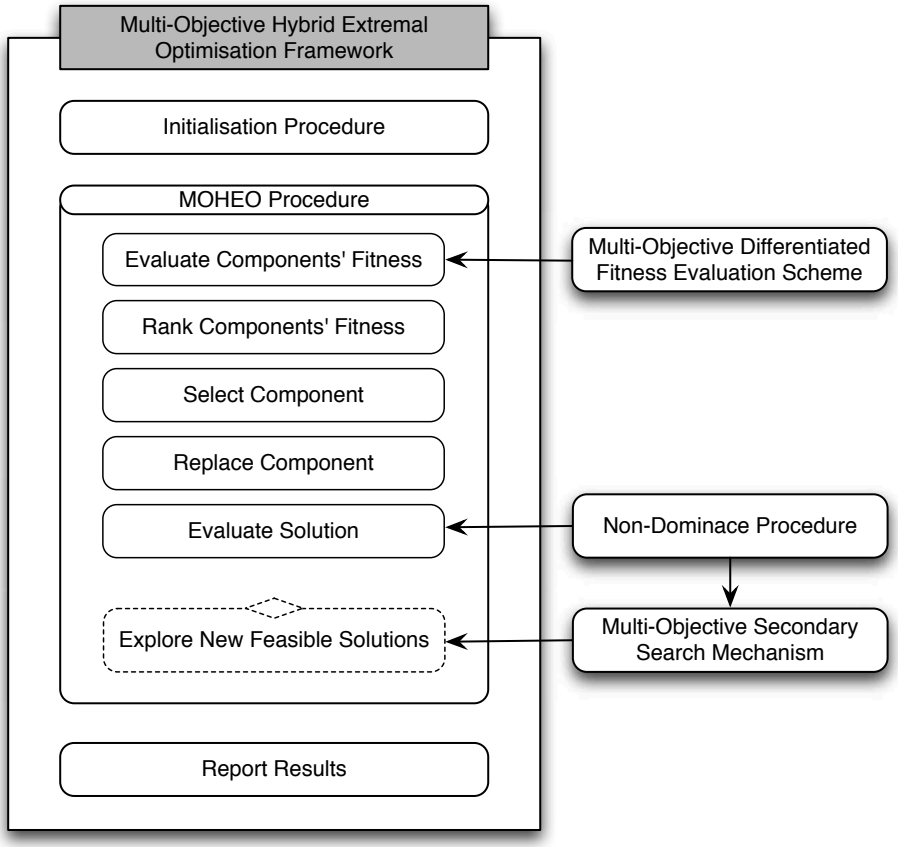


Figure 4.3: The Multi-Objective Hybrid Extremal Optimisation (MOHEO) Framework to solve multi-objective constrained combinatorial problems.

The main change in the initialisation process is to the action to be taken when the generated initial solution is feasible. Here, if the initial solution is feasible then this becomes the first point assigned to the approximated Pareto-front set found at the current iteration. Recall that in multi-objective optimisation the final result is composed of a collection of non-dominated solutions.

As was described in Section 4.2.1, the multi-objective differentiated fitness evaluation for each component of the solution is carried out through an aggregating function. This function takes each single-objective assessment of the components according to Equation 3.3 and integrates them into a single value as specified by Equation 4.1.

The rank components' fitness, select component and replace component modules in the multi-objective framework operate in the same way as the single-objective framework version.

Once the new solution has been generated, it is then evaluated to see whether it is feasible or not. In the case that a feasible solution is obtained, the non-dominance procedure is called to see if the new solution should be added to the approximated Pareto-front set found at the current iteration. If the new solution is added, the non-dominated procedure must verify if solutions exist in the current approximated Pareto-front that are dominated by the new solution, which must be eliminated. Otherwise, if an infeasible solution is generated then a new MOHEO iteration is performed.

The more radical change with respect to the single-objective framework lies in the activation of the secondary search mechanism. In this part, instead of activating the secondary search mechanism each time a feasible solution is found, it is triggered at pre-set intervals throughout the execution of the multi-objective hybrid extremal optimisation process in a periodical way. The implementation of this periodical activation is based on the fact that extremal optimisation performs one component's change in an iteration of the framework. In contrast, the secondary search mechanism could carry out an indeterminate number of component changes in the same iteration. For this reason, the framework gives the opportunity to the extremal optimisation part the necessary time to complete a considerable number of changes in the solution vector to converge towards the centre part of the Pareto-front surface. Initially, this period is defined by Formula 4.2.

$$b = \left(\frac{i}{\left(\frac{NI}{Pe}\right)} \bmod 2 \right) \tag{4.2}$$

where:

b	is a Boolean variable (0—1) that indicates if the secondary search mechanism is active (1) or not (0),
i	is the value for the current iteration,
NI	is the total number of iterations of the evolutionary process, and
Pe	is the parameter that specifies how many times the secondary search mechanism is activated and deactivated.

Note that this feature, in the secondary search mechanism, incorporates an additional parameter Pe to the MOHEO framework. For this reason, it is considered that in future research this parameter can be changed to a self-adaptive function, which

will determine when to activate or deactivate the secondary search mechanism in an autonomous way. In this thesis, the double traverse local search approach described in Algorithm 15 is invoked by the secondary search mechanism.

When the Boolean variable b is 0, then the extremal optimisation component has the opportunity to perform its search strategy alone, working without assistance. Hence, it concentrates its effects in the central part of the approximate Pareto-front set. However, when the Boolean variable b is 1, the extremal optimisation part works together with the double traverse local search as a combined strategy. Here, the solution starts the search towards the ends of the approximate Pareto-front set, which indirectly improves the convergence toward the real Pareto-front set. Figure 4.4 shows an expected approximated Pareto-front set when the aggregating function approximation and the double traverse local search approximation work together for a bi-objective maximisation problem.

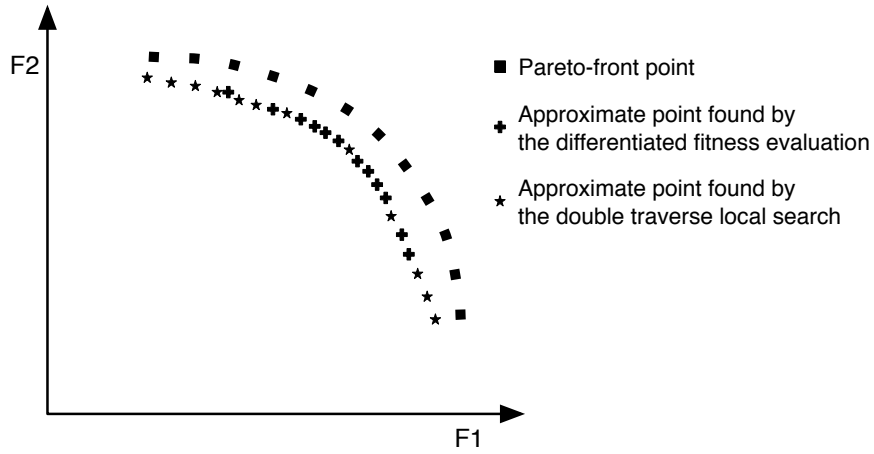


Figure 4.4: An illustrative approximated Pareto-front set, where F1 and F2 represent the two objective functions.

The *Non-Dominance* procedure is responsible for generating the archive with all non-dominated points found by the MOHEO process. Every time a feasible solution is found, by either the extremal optimisation search or the secondary search mechanism, this is sent to the non-dominated procedure. This procedure examines the new solution to decide whether, or not, this is a new non-dominated point to be added into the approximated Pareto-front set found at the current iteration. When a new non- dominated point is going to be incorporated to the approximated Pareto-front set, then it is necessary to eliminate all the points that become dominated by the new member. Thus, it is guaranteed that the set contains only non-dominated points.

Finally, the approximate Pareto-front set found by the MOHEO process is returned

in a file for later analysis. This analysis can be demonstrated by a graphic illustration or by some of the metrics that were presented in Section 2.5.2.

Algorithm 16 shows the pseudocode of the multi-objective hybrid extremal optimisation framework for multi-objective constrained combinatorial optimisation problems.

Algorithm 16 The MOHEO framework for multi-objective CCOPs.

1: {**Initialisation Procedure**}
2: Generate the probability vector P
3: Initialise solution S
4: **if** S feasible **then** Add S into the approximated Pareto-front set
5: {**HEO Procedure**}
6: **repeat**
7: **if** the current solution is feasible **then**
8: Evaluate the fitness $\Lambda(x_i)$ for the components of the feasible solution
9: **else**
10: Evaluate the fitness $\Lambda(x_i)$ for the components of the infeasible solution
11: **end if**
12: Rank the components according to its fitness $\Lambda(x_i)$ from the worst to the best
13: Select a component based on the probability of its rank P using any selection technique
14: Obtain a S_{new} in the neighbourhood of S when the selected component value is replaced by a different one
15: Evaluate the new solution S_{new}
16: **if** S_{new} is feasible **then**
17: Apply the $NonDominance(S_{new})$ procedure
18: {**Multi-Objective Secondary Search Mechanism Procedure**}
19: **if** the current iteration is in the defined period to perform the secondary search mechanism **then**
20: S_{new} = Apply the multi-objective secondary search mechanism to S_{new}
21: **end if**
22: **end if**
23: **until** the terminal condition is satisfied
24: **return** the approximated Pareto-optimal and Pareto-front sets

4.3 MOHEO Applied to Multi-Objective CCOPs

This section explains how the multi-objective hybrid extremal optimisation framework is applied to the multi-objective knapsack, the multi-objective quadratic assignment and the multi-objective permutation flow-shop scheduling problems. These problems were chosen because they are amongst the most well-studied problems and form the

basis of many different applications [69, 126, 168, 324]. For this reason, it is possible to find public benchmark problem instances for them with the necessary data to perform a proper comparison with other methods. These three problems with their respective benchmark problem instances will be used later to test the effectiveness of the new MOHEO approach.

Below, a concise description for each problem and the relevant aspects of its implementation is described.

4.3.1 Multi-Objective Knapsack Problem

The knapsack problem has been widely used as a benchmark problem to compare new general-purpose meta-heuristics. Moreover, there are several real-world problems that can be modelled like the knapsack problem, specially those that must assign a series of items linked with a profit and cost to a resource (knapsack) with limited capacity [289]. The idea is to maximise the profit without overloading the resource's capacity.

The multi-objective version of the classical 0/1 knapsack problems can be obtained by adding more knapsacks to the problem. Thus, each knapsack has an associated objective function and the solution vector applied to all objectives or knapsacks is the same. Many real-world applications can be found where a multi-objective knapsack representation has been used to solve a particular problem [50, 157, 164, 174, 296]. Formally, it can be represented as:

$$\text{Maximise } \vec{f}(\vec{x}) \quad = \quad \{f^1(\vec{x}), f^2(\vec{x}), \ldots, f^m(\vec{x})\} \tag{4.3}$$

with

$$\begin{aligned} f^j(\vec{x}) \quad &= \quad \sum_{i=1}^{n} p_{ij} x_i, \quad \forall j \quad 1 \leq j \leq m \\ \text{Subject to} \quad & \sum_{i=1}^{n} w_{ij} x_i \leq c_j, \quad \forall j \quad 1 \leq j \leq m \\ & x_i \in \{0,1\}, \quad \forall i \quad 1 \leq i \leq n \end{aligned}$$

where:

- p_{ij} is the profit of item i in knapsack j,
- w_{ij} is the weight of item i in knapsack j,
- c_j is the capacity of the knapsack j,

$\vec{x}$ is the solution vector $(x_1, x_2, \ldots, x_n)$, and

x_i is the i^{th} component in the solution vector, 1 if the item is in the solution, 0 otherwise.

4.3.1.1 Implementation

The implementation of the multi-objective knapsack problem takes as its basis some elements from the single-objective multi-dimensional knapsack problem implementation. This section describes the most relevant issues in this application so as to achieve a comprehensive understanding of it.

The representation of the solution for the multi-objective knapsack problem is the same as that used for the multi-dimensional knapsack problem illustrated in Figure 3.9. When the value 1 is assigned to a component this means that the item associated with this component is inside all knapsacks. On the other hand, when the value 0 is assigned to one component, it means that the item associated to this component is not considered to be put into the knapsacks. The initial solution is created through a random generation of binary values from the set $\{0, 1\}$ for each item. In the case of an initial feasible solution, this is saved in an archive as the first point in the approximated Pareto-front set.

Note that the difference between the multi-dimensional knapsack problem and the multi-objective knapsack problem is that the former consists of one objective function with m different constraints however the latter consists of m objective functions with only one constraint. Thus, the multi-objective fitness evaluation $\Lambda(x_i)$ is given by Equation 3.2, where the single-objective fitness evaluation $\lambda(x_i)$ is given by Equation 3.3, which is the same as the one used by the multi-dimensional knapsack problem. The difference in applying Equation 3.3 to one or another type of problem is just the number of constraints that must be considered when calculating the fitness.

When the solution is feasible, the components that potentially could be put in the knapsacks which keep the feasibility and contribute a high profit for each objective function are considered to be the components that degrade the current solution. This is because its absence prevents a better evaluation being obtained. Now, when the solution is infeasible, the components that potentially could be taken out from the knapsacks, eliminating the infeasibility and having a low profit, are considered as the components that degrade the current solution. This is because its presence ensures that the solution becomes infeasible.

When a feasible solution is found then the double traverse local search is called, on

the condition that the current generation is within an active period for the secondary search mechanism. Here, the multi-objective double traverse local search performs the swapping between items that are inside the knapsack with those that are outside. The pseudocode of the multi-objective local search interpretation for the multi-objective knapsack problem is described in Algorithm 17.

Algorithm 17 The multi-objective double traverse local search for the MOKP.

```
Generate a random variate vector P(m)_rand ~ U(1,0)
repeat
  for each item i within the knapsack do
    Select an objective using RWS and the P(m)_rand probability
    for each item j out of the knapsack do
      if the solution remains feasible when item i is exchanged by item j AND
      the evaluation of the new solution is increased then
        Implement the operation and update the variables that were involved
        if the new solution is non-dominated then
          Incorporate it into the approximated Pareto-front set found so far
        end if
      end if
    end for
  end for
  for each item i within the knapsack do
    Select an objective using RWS and the P(m)_rand probability
    for each item j and j' out of the knapsack do
      if the solution remains feasible when item i is exchanged by item j and j'
      AND the evaluation of the new solution is increased then
        Implement the operation and update the variables that were involved
        if the new solution is non-dominated then
          Incorporate it into the approximated Pareto-front set found so far
        end if
      end if
    end for
  end for
until there is no improvement in the solution
```

The multi-objective hybrid extremal optimisation procedure is repeated for a pre-set number of iterations. Finally, the approximated Pareto-optimal and the Pareto-front set that were found by the approach are returned as output. Algorithm 18 shows the multi-objective hybrid extremal optimisation pseudocode for the multi-objective knapsack problem.

Algorithm 18 The MOHEO framework for the multi-objective knapsack problem.

Generate the probability vector P
Initialise a solution S with random values from $\{0, 1\}$
if S feasible **then** Add S into the approximated Pareto-front set
repeat
  if the current solution is feasible **then**
    Evaluate the fitness $\Lambda(x_i)$ for the components of the feasible solution
  else
    Evaluate the fitness $\Lambda(x_i)$ for the components of the infeasible solution
  end if
  Rank the items according to its fitness $\Lambda(x_i)$ from the worst to the best
  Select an item based on the probability of its rank P using RWS
  Obtain a S_{new} in the neighbourhood of S changing the value of the selected item to 0 or 1 as appropriate
  Evaluate the new solution S_{new}
  if S_{new} is feasible **then**
    Apply the $NonDominance(S_{new})$ procedure
    if $\left(\frac{g}{\left(\frac{NI}{Pe}\right)} \bmod 2\right)$ **then**
      S_{new} = Apply the multi-objective secondary search mechanism to S_{new}
    end if
  end if
until the terminal condition is satisfied
return the approximated Pareto-optimal and Pareto-front sets

4.3.2 Multi-Objective Quadratic Assignment Problem

The Quadratic Assignment Problem (QAP) is a well-known $\mathcal{NP}$-hard optimisation problem [253] with a considerable number of real-world applications [48]. The quadratic assignment problem occurs in applications such as layout of malls, hospitals and airport terminals, location of electronic components in an integrated circuit, distribution of resources in a collection centre, and any other case where it is necessary to assign a set of facilities to specific locations. The objective is to minimise the cost associated with the flows of items amongst facilities and the distance between them.

Now, when there are two or more different sorts of flows amongst facilities, which must be assigned to a location, then the problem becomes a multi-objective quadratic assignment problem (MOQAP). For instance, a hospital could require the simultaneous minimisation of the costs associated with the flow of doctors, patients, nurses, visitors, and equipment between the different facilities, remembering that the distance amongst the location where the facilities are assigned must also be minimised too.

Formally, the multi-objective quadratic assignment problem can be represented as:

$$\text{Minimise } \vec{C}(\vec{\pi}) = \{C^1(\vec{\pi}), C^2(\vec{\pi}), \ldots, C^m(\vec{\pi})\}, \quad \bar{\pi} \in \Omega \tag{4.4}$$

with

$$C^k(\vec{\pi}) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b^k_{\pi_i \pi_j}, \quad 1 \leq k \leq m \tag{4.5}$$

where:

m	is the number of objectives or flows,
n	is the number of facilities/locations,
π_i	is the facility in the location i in permutation $\vec{\pi}$,
Ω	is the set of all permutations,
a_{ij}	is the distance between location i and location j, and
$b^k_{\pi_i \pi_j}$	is the k^{th} flow from facility assigned to location i to facility assigned to location j.

4.3.2.1 Implementation

An integer vector can represent the solution for the multi-objective quadratic assignment problem, where the index of the vector indicates the location number and the value stored in each component of the vector denotes the number of the facility that was assigned to the respective location. The different possible solutions are represented by the permutations of facilities between locations. Figure 4.5 illustrates the representation of a solution for the multi-objective quadratic assignment problem.

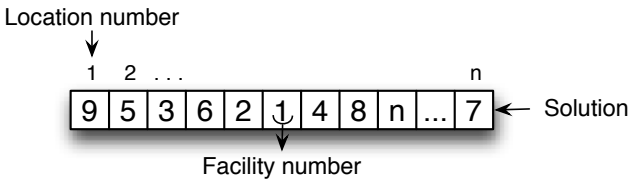


Figure 4.5: An example of a solution vector for the MOQAP.

The initial solution is generated by a random permutation that is directly added as the first point in the approximated Pareto-front set. Note that the permutation representation for the multi-objective quadratic assignment problem only generates feasible solutions. For this reason, the separated fitness evaluation scheme will be applied only using the branch when the current solution is feasible.

The differentiated fitness evaluation is applied to identify which facilities, that

have been assigned to different locations, degrade the solution. Taking as reference Equation 3.2 and considering that there is no constraint that may be violated or satisfied (V_i or S_i), only the component w_i that is related to a profit or cost coefficient is considered. Equation 4.6 illustrates the single-objective fitness evaluation for the feasible case based on the flow from one facility to the rest of them.

$$\lambda^k(x_i) = \begin{cases} -\sum_{j=1}^{n} a_{ij} b_{\pi_i \pi_j} & \text{for feasible solutions} \\ \\ NULL & \text{for infeasible solutions} \end{cases} \tag{4.6}$$

where:

n	is the number of facilities/locations,
π_i	is the facility assigned to location i in permutation $\vec{\pi}$,
a_{ij}	is the distance between location i and location j,
$b_{\pi_i \pi_j}$	is the flow from facility assigned to location i to facility assigned to location j, and
$\lambda^k(x_i)$	is the k^{th} single-objective fitness function applied to the i^{th} component.

Thus, each time a new solution is generated, the fitness function looks for the facilities that contribute to the larger flow/distance costs, because these are considered as the components that degrade the solution. This is due to their current assignments preventing the solution from achieving a better evaluation, as the aim is to minimise the objective function.

The roulette wheel technique is used to select one of the worst evaluated facilities according to its probability P. The chosen facility is permuted with other randomly selected facility. This means that they interchange the location where they were previously allocated.

Next, the double traverse local search is called on the condition that the current generation is within an active period for the secondary search mechanism. For the multi-objective quadratic assignment problem, the double traverse local search is applied by carrying out swaps between the facilities that are assigned to different locations. By the fact that this problem works with a permutation representation of the solution, the second part of the double traverse local search algorithm is not applicable. Algorithm 19 presents the double traverse local search pseudocode adapted to work with the multi-objective quadratic assignment problem.

Algorithm 19 The multi-objective double traverse local search for the MOQAP.

```
Generate a random variate vector P(m)_rand ~ U(1,0)
repeat
  for each location i do
    Select an objective using RWS and the P(m)_rand probability
    for each location j do
      if the evaluation of the new solution is decreased when the facility assigned
      to location i is swapped with the facility assigned to location j then
        Implement the operation and update the variables that were involved
        if the new solution is non-dominated then
          Incorporate it into the approximated Pareto-front set found so far
        end if
      end if
    end for
  end for
until there is no improvement in the solution
```

The multi-objective hybrid extremal optimisation procedure is repeated for a pre-set number of iterations. Finally, the approximated Pareto-optimal and the Pareto-front set that were found by the approach are returned as output. Algorithm 20 shows the multi-objective hybrid extremal optimisation pseudocode for the multi-objective quadratic assignment problem. Note that the conditions to verify the feasibility of the solution are not present, as the representation of the solution only generates feasible solutions.

Algorithm 20 The MOHEO framework for the MOQAP.

```
Generate the probability vector P
Initialise a solution S with random assignment of facilities to locations
Add S into the approximated Pareto-front set
repeat
  Evaluate the fitness Λ(x_i) for the components of the solution
  Rank the facilities according to its fitness Λ(x_i) from the worst to the best
  Select a facility based on the probability of its rank P using RWS
  Obtain a S_new in the neighbourhood of S, swapping the location of the selected
  facility with the location of another facility randomly chosen
  Evaluate the new solution S_new
  Apply the NonDominance(S_new) procedure
  if ( g / (NI/Pe) mod 2 ) then
    S_new = Apply the multi-objective secondary search mechanism to S_new
  end if
until the termination condition is satisfied
return the approximated Pareto-optimal and Pareto-front sets
```

4.3.3 Multi-Objective Permutation Flow-Shop Scheduling Problem

The Multi-Objective Permutation Flow-Shop Scheduling Problem (MOPFSSP) is one of the variants of the Flow-Shop problems and it is one of the most commonly studied scheduling problems [108]. Some of the multiple objectives that can be applied to this problem are the optimisation of the makespan, the total flow time, the maximum tardiness, and the total tardiness. This problem has a search space of size $n!$ making it in an $\mathcal{NP}$-hard problem, where n is the number of jobs.

The permutation flow-shop scheduling problem consists of the allocation of a set of N jobs to be processed by a set of M machines. All machines perform the same sequences or schedules of jobs that are represented as a permutation $\pi = \{\pi_1, \ldots, \pi_n\}$ and all jobs are processed by machines that are technologically identical. The machines are visited for all jobs in the same sequence from M_1 to M_m. The main constraint in this problem is that the j^{th} job assigned to the i^{th} machine can only start its processing when both the j^{th} job has finished its processing in the $(i-1)^{th}$ machine and the i^{th} machine is available. The flow-shop scheduling can be modelled by a completion time matrix $C(j,i)$ that can be filled following Equations 4.7 to 4.10.

$$C(\pi_1, 1) = P(\pi_1, 1) \tag{4.7}$$

$$C(\pi_j, 1) = C(\pi_{j-1}, 1) + P(\pi_j, 1) \quad 2 \leq j \leq n \tag{4.8}$$

$$C(\pi_1, i) = C(\pi_1, i-1) + P(\pi_1, i) \quad 2 \leq i \leq m \tag{4.9}$$

$$C(\pi_j, i) = \max\{C(\pi_{j-1}, i), C(\pi_j, i-1)\} + P(\pi_j, i) \quad 2 \leq j \leq n, \quad 2 \leq i \leq m \tag{4.10}$$

where:

n	is the number of jobs,
m	is the number of machines,
$C(\pi_j, i)$	is the completion time for the j^{th} scheduled job performed in the i^{th} machine,
$P(\pi_j, i)$	is the processing time of the j^{th} scheduled job in the i^{th} machine.

The completion time for a job is C_j is given when the job has been processed by the last machine, that is $C_j = C_{j,m}$.

The objective functions are given by the minimisation of the maximum completion time or makespan $C_{\max}$, the minimisation of the maximum tardiness $T_{\max}$ and the

minimisation of the sum of the completion times C_{sum} represented by Equations 4.11, 4.12 and 4.13.

$$C_{\max} = \max\{C_j | j = 1, 2, ..., n\} \quad (4.11)$$

$$T_{\max} = \max T_j = \{\{(C_j - D_j), 0\} | j = 1, 2, ..., n\} \quad (4.12)$$

$$C_{sum} = \sum_{j=1}^{n} C_j \quad (4.13)$$

where:

n	is the number of jobs, and
$D(j)$	is the due date for the j^{th} job.

4.3.3.1 Implementation

The solution for the multi-objective permutation flow-shop scheduling problem can be represented by an integer vector, which describe the sequence of n jobs that are going to be processed by m machines. Each job is distinguished by its identification number. The different possible sequences of jobs or solutions are represented by any permutation of the n jobs. Figure 4.6 illustrates the representation of the solution for the multi-objective permutation flow-shop scheduling problem.

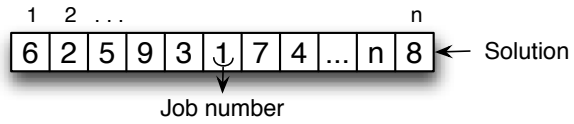


Figure 4.6: An example of a solution vector for the MOPFSSP.

The initial solution is generated by a random permutation of jobs. Like the multi-objective quadratic assignment problem the permutation representation for the multi-objective permutation flow-shop scheduling problem only generates feasible solutions. This means that the initial solution is directly added as the first point in the approximated Pareto-front set. The separated fitness evaluation scheme will only be applied using the branch when the current solution is feasible.

The differentiated fitness evaluation for this multi-objective problem has an important difference with respect to the previous two multi-objective problems. Until now, the objectives to be maximised or minimised have been similar to each other; that is,

they are described by the same concept. In the case of the multi-objective knapsack problems, each objective is described as a knapsack; likewise, in the multi-objective quadratic assignment problem, each objective is described as a flow. However, for the multi-objective permutation flow-shop scheduling problem, the objectives are described by different concepts such as the completion time or makespan $C_{\max}$, the tardiness $T_{\max}$ and the sum of the completion times C_{sum}.

In this thesis, the bi-objective instances for the multi-objective permutation flow-shop scheduling problem will be considered. One objective is related to minimise the maximum makespan $C_{\max}$ and the other objective is associated with minimising the maximum tardiness $T_{\max}$. These objective functions are not similar. For this reason, Equation 3.2 evaluates each of these objectives in a different way.

Furthermore, this problem has a couple of features that are important to highlight. First, like in the multi-objective quadratic assignment problem, there is no constraint that may be violated or satisfied (V_j or S_j). Additional to that, there is not a profit or cost coefficient to be considered as the w_i component. However, it is possible to know the makespan C_j and tardiness T_j values per job, which are used to evaluate both objective functions. Thus, Equations 4.14, 4.15 and 4.16 illustrate the differentiated fitness evaluation that is applied to identify which jobs degrade the solution.

$$\lambda'(x_j) = \begin{cases} -\overline{C}_j & \text{for feasible solutions} \\ \\ NULL & \text{for infeasible solutions} \end{cases} \tag{4.14}$$

$$\lambda''(x_j) = \begin{cases} -\overline{T}_j & \text{for feasible solutions} \\ \\ NULL & \text{for infeasible solutions} \end{cases} \tag{4.15}$$

$$\Lambda(x_j) = \lambda'(x_j) + \lambda''(x_j) \quad ,\forall j \quad 1 \leq j \leq n \tag{4.16}$$

where:

$\overline{C}_j$	is the normalised makespan for the j^{th} job,
$\overline{T}_j$	is the normalised tardiness for the j^{th} job, and
$\Lambda(x_j)$	is the multi-objective fitness of the j^{th} job.

The fitness function looks for the jobs that contribute to the larger makespan and tardiness values as these are considered to be the components that degrade the solution. As the aim is to minimise the objective function, the current assignment for these components prevents the solution achieving a better evaluation.

The roulette wheel technique is used to select one of the worst evaluated jobs according to its probability P. The chosen job is permuted with other randomly selected jobs. The new solution is evaluated and then examined by the non-dominance function to verify if this is a new non-dominated point.

After this, the double traverse local search is called on the condition that the current generation is within an active period for the secondary search mechanism. For the multi-objective permutation flow-shop scheduling problem, the double traverse local search is applied by carrying out swaps between the jobs. Due to the permutation representation, like in the multi-objective quadratic assignment problem, only the simple swap part is performed. Algorithm 21 presents the double traverse local search pseudocode adapted to work with the multi-objective permutation flow-shop scheduling problem.

Algorithm 21 The multi-objective double traverse local search for the MOPFSSP.

```
1: Generate a random variate vector P(m)_rand ~ U(1,0)
2: repeat
3:   for each job i do
4:     Select an objective using RWS and the P(m)_rand probability
5:     for each job j do
6:       if the evaluation of the new solution is decreased when job i is swapped
         with job j then
7:         Implement the operation and update the variables that were involved
8:         if the new solution is non-dominated then
9:           Incorporate it into the approximated Pareto-front set found so far
10:        end if
11:      end if
12:    end for
13:  end for
14: until there is no improvement in the solution
```

The multi-objective hybrid extremal optimisation procedure is repeated for a preset number of iterations. Finally, the approximated Pareto-optimal and the Pareto-front set that were found by the approach are returned as output. Algorithm 22 shows the multi-objective hybrid extremal optimisation pseudocode for the multi-objective permutation flow-shop scheduling problem. Note, like the quadratic assignment prob-

lem, the conditions to verify the feasibility of the solution are not present, as the representation of the solution generates only feasible solutions.

Algorithm 22 The MOHEO framework for the MOPFSSP.

1: Generate the probability vector P
2: Initialise a solution S with random assignment of jobs
3: Add S into the approximated Pareto-front set
4: **repeat**
5: Evaluate the fitness $\Lambda(x_i)$ for the components of the solution
6: Rank the jobs according to their fitness $\Lambda(x_i)$ from the worst to the best
7: Select a job based on the probability of its rank P using RWS
8: Obtain a S_{new} in the neighbourhood of S, swapping the selected job with another job randomly chosen
9: Evaluate the new solution S_{new}
10: Apply the $NonDominance(S_{new})$ procedure
11: **if** $\left(\frac{g}{\left(\frac{NI}{Pe}\right)} \bmod 2\right)$ **then**
12: S_{new} = Apply the multi-objective secondary search mechanism to S_{new}
13: **end if**
14: **until** the termination condition is satisfied
15: **return** the approximated Pareto-optimal and Pareto-front sets

4.4 Computational Experiments

The proposed multi-objective hybrid extremal optimisation algorithm was coded in the C language and compiled with `gcc`. The computing platform used to perform the tests had a 1.86 GHz Intel Core2 CPU, 917 MB of memory and ran under a Linux OS.

Each problem instance was run ten times. For each run the random seed was changed to a new value. The number of iterations to complete a multi-objective hybrid extremal optimisation process was 100000. The tests performed for each problem with a large number of iterations reported no significant improvements in results. The τ parameter was tested within the range of values between 1.1 and 2.8 in increments of 0.1. For values outside this range of τ the results were not competitive. As a result, τ was set at 1.4 for all instance problems, which gave the best overall results. The same τ has been reported in previous research [42, 143, 243] as being a good value to obtain efficient solutions.

The three basic concepts to be considered in this section for multi-objective optimisation are:

1. *Minimise the convergence* - that is the Pareto-front distance produced by this proposal with respect to the true Pareto-front (assuming that it is known).

2. *Maximise the diversity* - that is the distribution of solutions, so that a Pareto-front distribution which is as uniform as possible can be produced.

3. *Maximise the coverage* - that is to extend the Pareto-front solutions towards the furthest areas of the landscape.

The results are presented by means of plotted graphics and by using some on the metrics described in Section 2.5.2.

4.4.1 Test Data for the MOKP

In this first experiment, the multi-objective hybrid extremal optimisation algorithm is applied to solve a group of six different multi-objective knapsack instance problems from Zitzler and Thiele [324]. The test instance problems and test data were obtained from Zitzler and Laumanns [320].

The test data was generated with the following features. The profit and weight arrays are formed by uncorrelated random integers in the interval [10,100]. The knapsack capacity is set to half the total weight of all items with respect to a particular knapsack. That is:

$$c_i = 0.5 \sum_{j=1}^{m} w_{ij}$$

The name of the problem is presented through the nomenclature MOKP.$nnn.m$ where nnn is the number of items and m is the number of objectives.

4.4.1.1 Results

For the first four problems, MOKP.100.2, MOKP.250.2, MOKP.500.2 and MOKP.100.3, the results are compared with the true Pareto-optimal set (POS), which is available in Zitzler and Laumanns [320]. For the last two problems, MOKP.750.2 and MOKP.750.3, the results are compared with those obtained by the Strength Pareto Evolutionary Algorithm 2 (SPEA2) and Non-Dominated Sorting Genetic Algorithm II (NSGA-II). The reason for using this particular selection of problems and benchmarks of comparison was due to the availability of experimental results in Zitzler and Laumanns [320].

Below, Figures 4.7, 4.8 and 4.9 show the plotted graphs for the results obtained with the bi-objective knapsack instances MOKP.100.2, MOKP.250.2 and MOKP.500.2 respectively. Figures 4.10 and 4.11 show the plotted graphs for the results obtained with the tri-objective knapsack instances MOKP.100.3.

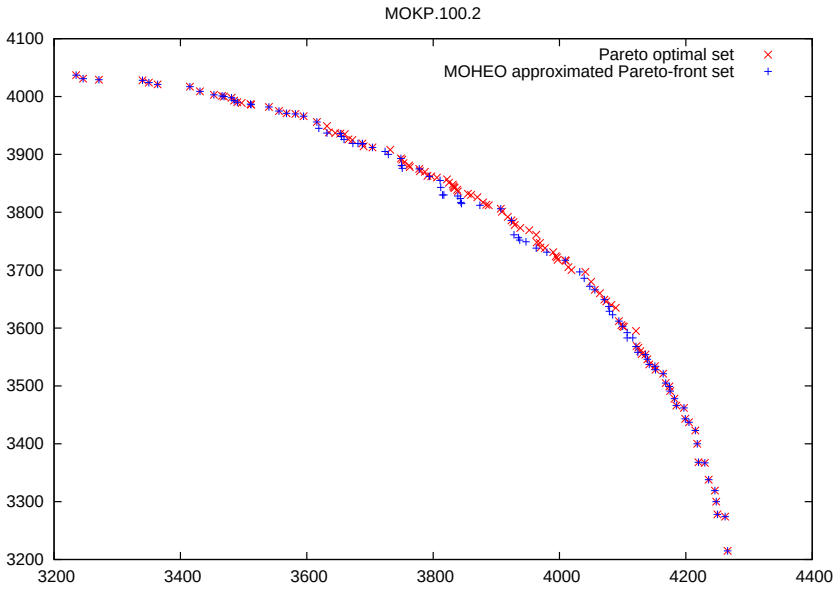


Figure 4.7: The MOHEO result of a single typical run for the bi-objective MOKP test problems with 100 items versus the true POS.

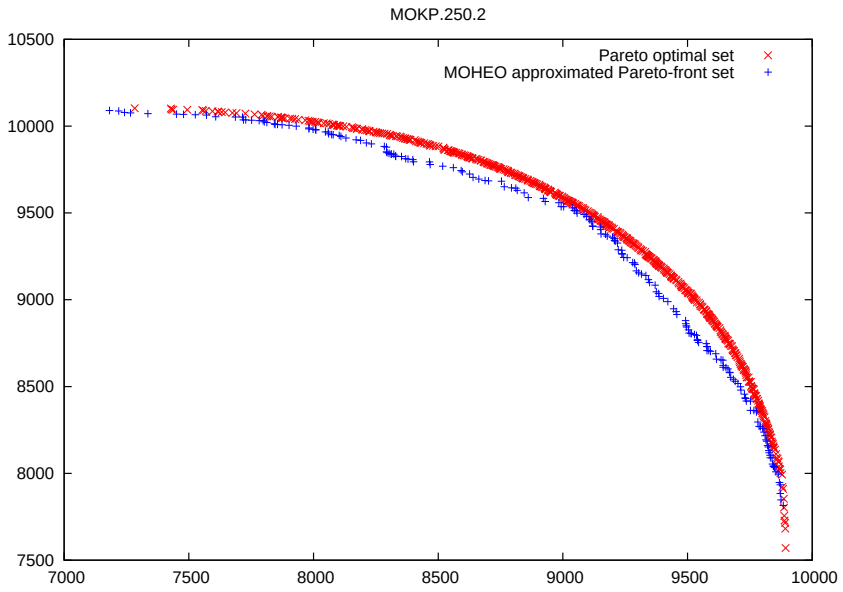


Figure 4.8: The MOHEO result of a single typical run for the bi-objective MOKP test problems with 250 items versus the true POS.

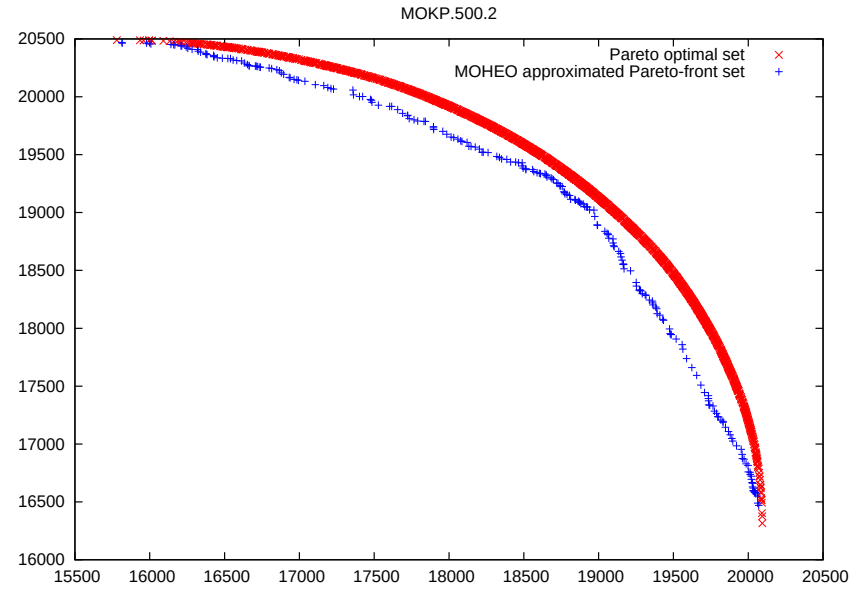


Figure 4.9: The MOHEO result of a single typical run for the bi-objective MOKP test problems with 500 items versus the true POS.

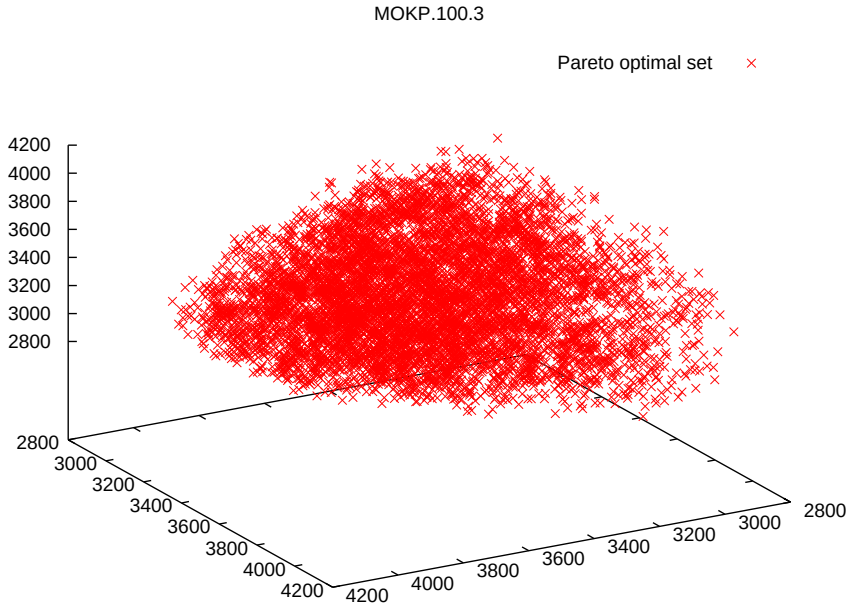


Figure 4.10: The Pareto-optimal set for the three objective MOKP test problems with 100 items.

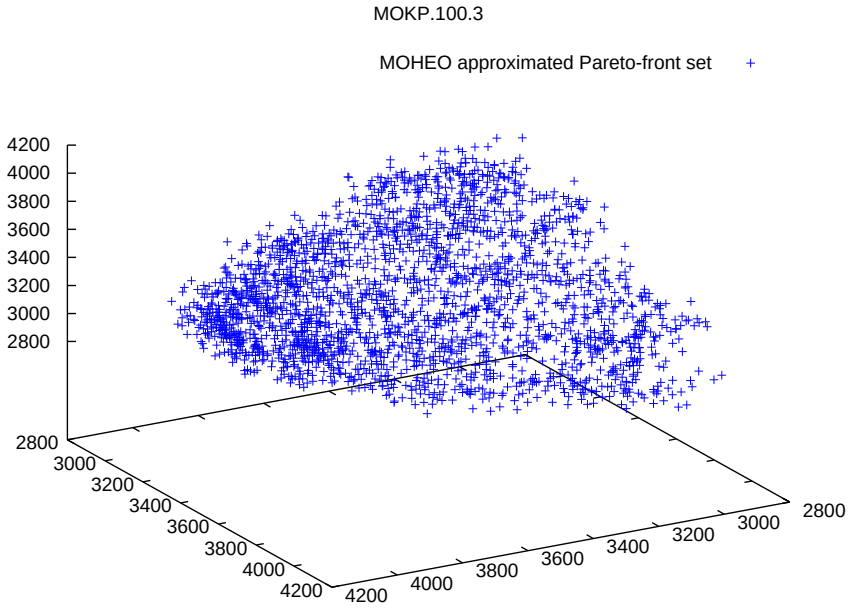


Figure 4.11: The MOHEO result of a single typical run for the three objective MOKP test problems with 100.

Next, Figure 4.12 shows the plotted graph with the results for the bi-objective knapsack instance MOKP.750.2 and Figures 4.13, 4.14 and 4.15 show the plotted graphs with the results for the tri-objective knapsack instance MOKP.750.3.

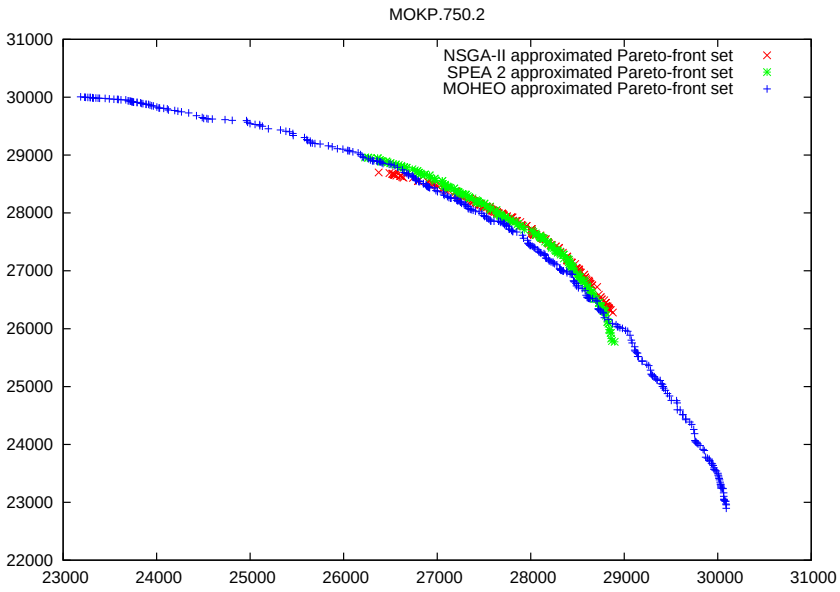


Figure 4.12: The MOHEO result of a single typical run for the bi-objective MOKP test problems with 750 items versus SPEA2 and NSGA-II.

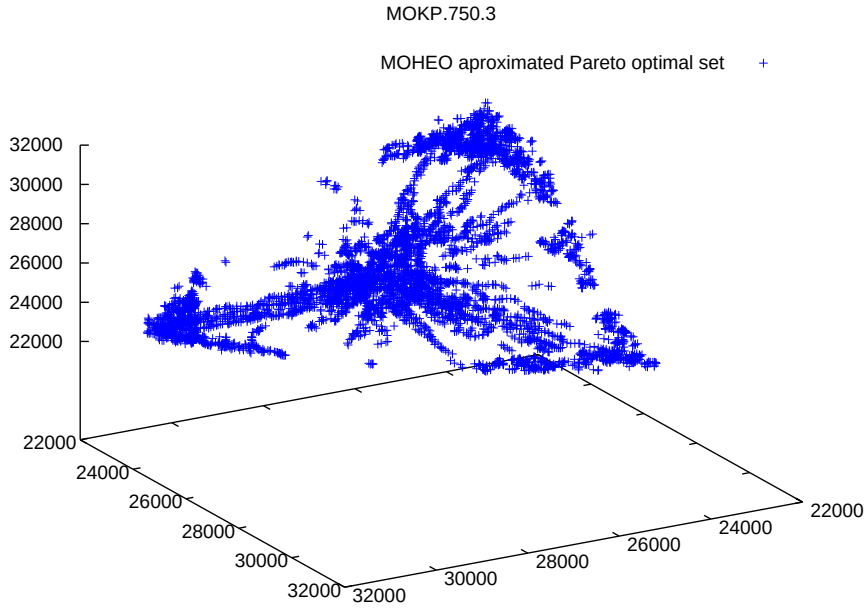


Figure 4.13: The MOHEO result of a single typical run for the three objective MOKP test problems with 750 items.

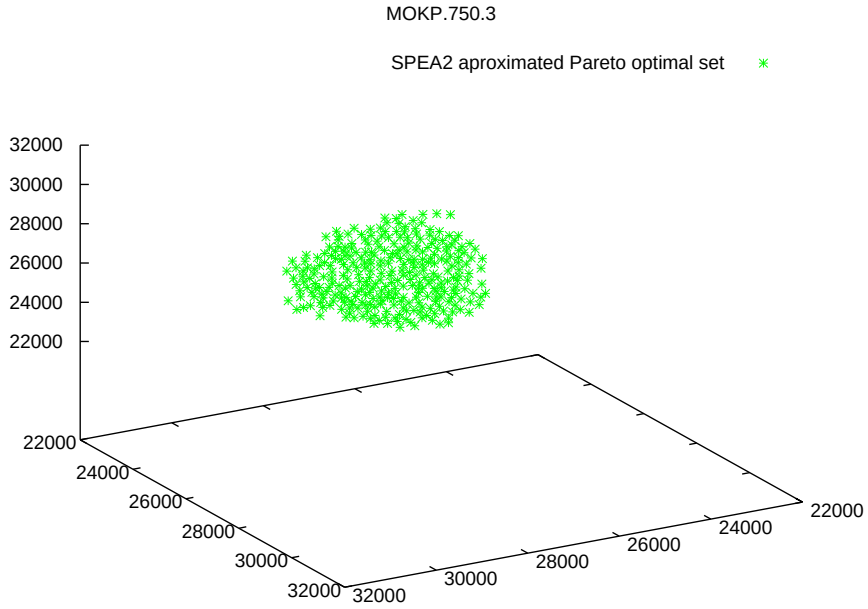


Figure 4.14: The SPEA2 result of a single typical run for the three objective MOKP test problems with 750 items.

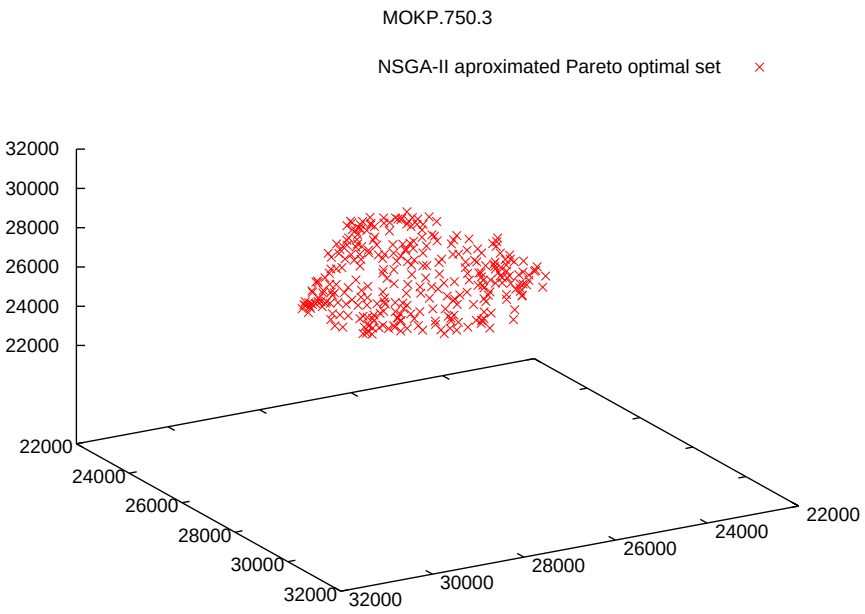


Figure 4.15: The NSGA-II result of a single typical run for the three objective MOKP test problems with 750 items.

Tables 4.9 and 4.10 present the S-metric and C-metric values respectively for the results obtained for the multi-objective knapsack instance MOKP.750.2 and MOKP.750.3.

Problem name	MOKP.750.2		MOKP.750.3	
	$\bar{\chi}$	σ	$\bar{\chi}$	σ
MOHEO	3.86941×10^7	5.37987×10^4	2.33542×10^{11}	5.11311×10^8
SPEA2	3.48117×10^7	3.54090×10^5	1.91337×10^{11}	1.94082×10^9
NSGA-II	3.34971×10^7	4.05495×10^5	1.82142×10^{11}	2.07875×10^9

Table 4.9: The S-metric values for the MOKP.750.2 and MOKP.750.3 test problems. The values are the averages and the standard deviations for the 10 runs.

Problem name	MOKP.750.2			MOKP.750.3		
	MOHEO	SPEA2	NSGA-II	MOHEO	SPEA2	NSGA-II
MOHEO	-	0.0828	0.02264	-	0.5977	0.87957
SPEA2	0.45518	-	0.17572	0.07739	-	0.7961
NSGA-II	0.40276	0.64536	-	0.00872	0.0169	-

Table 4.10: The C-metric values for the MOKP.750.2 and MOKP.750.3 test problems. The values are the averages for the 10 runs.

4.4.1.2 Discussion

With respect to the multi-objective knapsack test problems MOKP.100.2, MOKP.250.2, MOKP.500.2 (see Figures 4.7, 4.8 and 4.9), and MOKP.100.3 (see Figure 4.11) whose true POS results are known (see Figure 4.10), it can be observed that the most successful feature achieved by MOHEO is the coverage followed by the diversity and finally the convergence. For the four test problems, the plotted graphs show that the solutions obtained by MOHEO achieve quite similar coverage when compared to the true POS. In addition, the solutions present a regular distribution of points along the approximated Pareto-front set for the problems with two or three objectives. However, the number of solutions that were found are less than those in the true POS.

It is important to note that MOHEO showed degradation in the convergence as the number of variables was increased from 100 to 500 items, which can be interpreted as MOHEO manifesting certain sensitivity to the number of items that can be processed.

For the problems MOKP.750.2 (see Figure 4.12) and MOKP.750.3 (see Figure 4.13), which are compared against the well known methods SPEA2 and NSGA-II (see Figures 4.14 and 4.15), it can be observed again that the most successful feature achieved by MOHEO is the coverage. By looking at the figures it may be seen that in both test problems the area or surface covered by the approximated Pareto-front set obtained with the proposed algorithm is broader than those found by SPEA2 and NSGA-II.

Diversity worked quite well for the test problem with two objectives, but the test problem with three objectives shows some surface areas without solutions. Now, when the distribution of points between test problems MOKP.100.3 and MOKP.750.3 was compared, it can be inferred that the greater the number of items, the greater the difficulty to achieve a uniform distribution of points.

It is interesting to note that the convergence seems to be the weak point of the implementation to solve the multi-objective knapsack problem through the MOHEO framework. Despite reaching an attainment surface very close to those achieved by SPEA2 and NSGA-II, only a few points dominate those of SPEA2.

The test problems MOKP.750.2 and MOKP.750.3 can be analysed in a better way by presenting some metrics that help to reinforce the information presented in Figures 4.12, 4.13, 4.14 and 4.15. Thus, the S-metric and C-metric proposed by Zitzler [318] are given. The S-metric measures how much of the objective space is dominated by a given non-dominated Pareto-front set A. This metric is useful to measure the coverage of a solution in an independent way. A high value of this metric means a wide coverage of the Pareto-front set under analysis. On the other hand, using the C-metric (coverage) two Pareto-front sets can be compared to each other.

For this, the nomenclature used is $C(A, B)$ and the interpretation is as follows. The value $C(A, B) = 1$ means that all solutions in B are dominated by A. The value $C(A, B) = 0$ represents the situation where none of the solutions in B are dominated by A. Both orderings have to be considered since $C(A, B)$ is not necessarily equal to $1 - C(B, A)$. A more detailed explanation of these metrics can be found in Knowles and Corne [166].

Table 4.9 shows that MOHEO achieves a better coverage of the dominated space on MOKP.750.2 and MOKP.750.3. These results concur with Figures 4.12, 4.13, 4.14 and 4.15. In the test problem MOKP.750.2, Table 4.10 indicates that MOHEO dominates only a reduced number of solutions of SPEA2 and NSGA-II. However, 40% to 45% of solutions from MOHEO are not dominated by either of the two other methods (SPEA2, NSGA-II). In the case of MOKP.750.3, the table shows that MOHEO dominates around 60% of the solutions of SPEA2 and 88% of NSGA-II, and less than 1% of MOHEO solutions are dominated by either SPEA2 or NSGA-II. The latter shows a significantly better performance of MOHEO above SPEA2 or NSGA-II.

Finally, the inferior results obtained in convergence by MOHEO suggest that either the criterion to calculate the fitness or the secondary search mechanism was not the most appropriate to be applied for the multi-objective knapsack problem.

4.4.2 Test Data for the MOQAP

In the second experiment, the multi-objective hybrid extremal optimisation framework is applied to solve a group of eight different multi-objective quadratic assignment instance problems proposed by Knowles and Corne [168]. These test problems have ten facilities/locations with two objectives that correspond to the two flow matrices. Three out of eight instances were created using a uniformly random instance generator and the other five instances were created using a real-life instance generator. The name of each instance is presented through the nomenclature KC*nn*-*m*fl-*itype* where *nn* is the number of facilities/locations, m is the number of objectives or flows and *itype* is the instance identification starting with a correlative number plus the instance generator used (*uni* for uniformly random and *rl* for real-life distribution, which are defined in Knowles and Corne [168]).

4.4.2.1 Results

The test instances and test data were obtained from Knowles [165]. The results are compared with the true Pareto-optimal set (POS), which is available from this source.

The reason to use this particular selection of problems and benchmarks of comparison was due to the availability of experimental results from this source.

Below, Figures 4.16, 4.17 and 4.18 show the plotted graphs for the results obtained with the MOQAP instances KC10-2fl-3uni, KC10-2fl-1rl and KC10-2fl-5rl respectively.

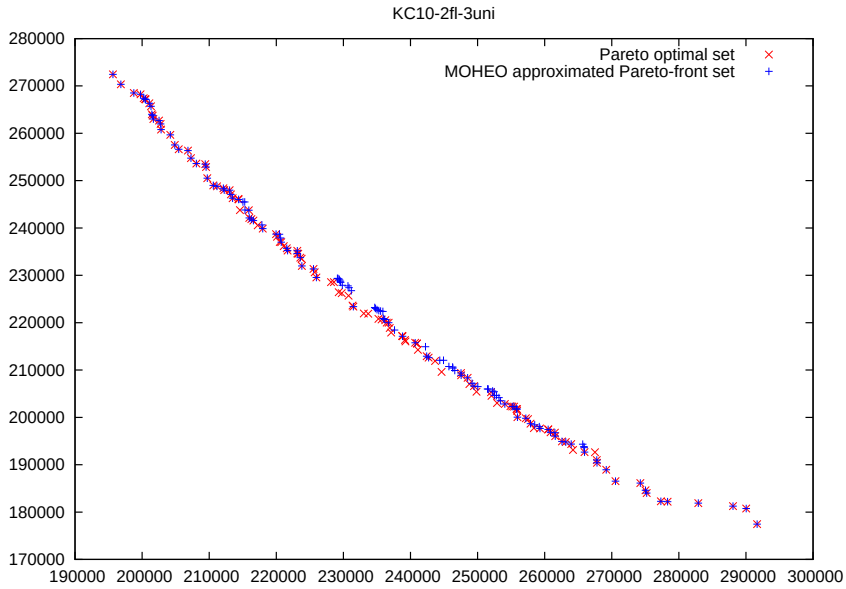


Figure 4.16: The MOHEO result of a single typical run for the bi-objective MOQAP (KC10-2fl-3uni) test problem with uniformly random distribution.

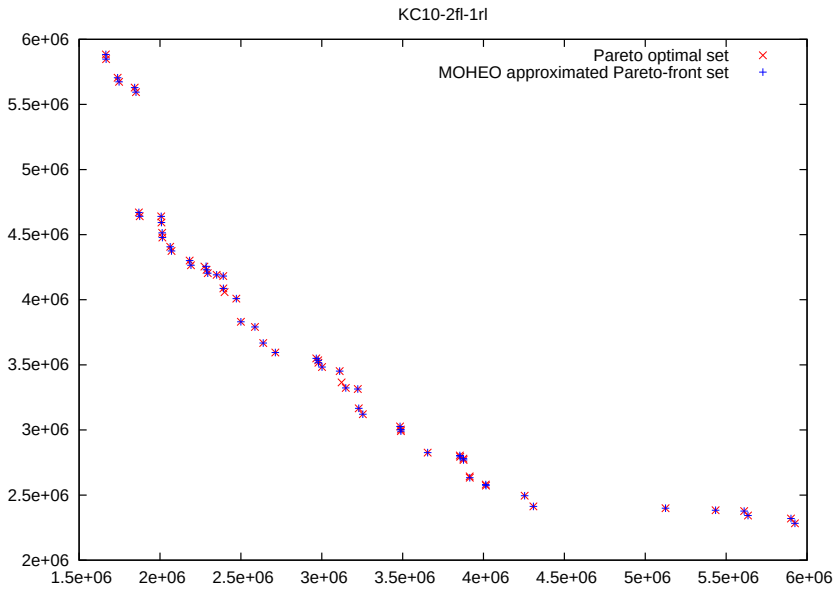


Figure 4.17: The MOHEO result of a single typical run for the bi-objective MOQAP (KC10-2fl-1rl) test problem with real-life distribution.

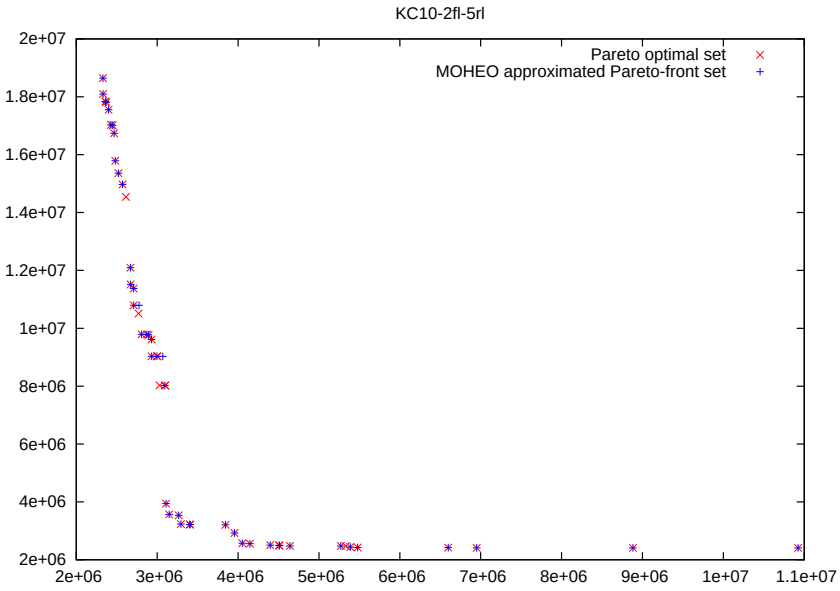


Figure 4.18: The MOHEO result of a single typical run for the bi-objective MOQAP (KC10-2fl-5rl) test problem with real-life distribution.

Besides, Tables 4.11 and 4.12 present the S-metric and C-metric values with the results obtained for the multi-objective quadratic assignment instances KC10-2fl-3uni, KC10-2fl-1rl and KC10-2fl-5rl.

Problem name	KC.10.2fl-3uni	
	$\bar{\chi}$	σ
MOHEO	5.358×10^9	6.407×10^6
TruePOS	5.382×10^9	0

Problem name	KC.10.2fl-1rl		KC.10.2fl-5rl	
	$\bar{\chi}$	σ	$\bar{\chi}$	σ
MOHEO	1.162×10^{13}	1.893×10^9	1.305×10^{14}	3.739×10^{10}
TruePOS	1.162×10^{13}	0	1.307×10^{14}	0

Table 4.11: The S-metric values for the KC.10.2fl-3uni, KC.10.2fl-1rl and KC.10.2fl-5rl test problems. The values are the averages and the standard deviations for ten runs.

Problem name	KC.10.2fl-3uni		KC.10.2fl-1rl		KC.10.2fl-5rl	
	MOHEO	TruePOS	MOHEO	TruePOS	MOHEO	TruePOS
MOHEO	-	0	-	0	-	0
TruePOS	0.245924	-	0.014232	-	0.06605	-

Table 4.12: The C-metric values for the KC.10.2fl-3uni, KC.10.2fl-1rl, and KC.10.2fl-5rl test problems. The values are the averages for the ten runs.

4.4.2.2 Discussion

From the experiments performed for the multi-objective quadratic assignment problem it can be said that in five out of eight instance problems, MOHEO successfully achieved the true POS. These test problems are KC.10.2f.1uni, KC.10.2f.2uni, KC.10.2f.2rl, KC.10.2f.3rl and KC.10.2f.4rl. The remaining three problems KC.10.2f.3uni (see Figure 4.16), KC.10.2f.1rl (see Figure 4.17), and KC.10.2f.5rl (see Figure 4.18), MOHEO reached solutions very close to the true POS. For this reason, C-metric and S-metric were used again to quantify the performance of these three problems.

From the S-metric measure shown in Table 4.11 and Figures 4.16, 4.17 and 4.18 it can be seen that the difference in the size of the dominated space between the results obtained by the multi-objective hybrid extremal optimisation approach and the true POS is minimal in favour of the true POS. In this case, the convergence, diversity, and coverage features worked quite well. A similar situation can be observed when the C-metric data in Table 4.12 was analysed. For the two real-life test problems the percentage of solutions dominated by the true POS was less than 7%. The only test problem with relatively poor quality results was KC.10.2fl-3uni, where the percentage of solutions dominated by the true POS was around 25%.

4.4.3 Test Data for the MOPFSSP

In the third experiment, the multi-objective hybrid extremal optimisation framework was applied to solve a set of instances for the multi-objective permutation flow-shop scheduling problem. These instances are proposed by Ishibuchi, Yoshida and Murata [154] and consist of eight tests. These eight tests are divided into two groups, the first with four bi-objective instances and the second with four tri-objective instances. In this thesis, the first group of four bi-objective instances are solved and the second group will be carried out in a future work.

The instances are defined as n-jobs and m-machines problems and can be described as follows. The processing time of each job on each machine was specified as a random integer in the interval $(1, 99)$. The due date of each job was specified by adding a random integer in the interval $(-100, 100)$ to its actual completion time in a randomly generated schedule. All test problems have 20 machines (i.e. $m = 20$). The number of jobs n can take one of the following values $n = 20, 40, 60, 80$. The number of objectives k is identified by the values $k = 2, 3$.

For the bi-objective tests (i.e., $k = 2$), to minimise the makespan and the max-

imum tardiness were evaluated. The name of the problem is presented through the nomenclature nnJmmMkO where nn is the number of jobs, mm is the number of machines and k is the number of objectives.

4.4.3.1 Results

The test instances for the multi-objective permutation flow-shop scheduling problem with their respective test data were obtained from Ishibuchi [152]. The obtained results by MOHEO are compared with the reference Pareto-front set, which is also available from this source. This reference Pareto-front set is made up of the all non-dominated points found from the SPEA 2, NSGA-II and MOGLS [154] methods. The reason to use this particular selection of problems and benchmarks of comparison was subject to the availability of experimental results from the source.

Below, Figures 4.19, 4.20, 4.21 and 4.22 show the plotted graphs for the results obtained with the multi-objective permutation flow-shop scheduling problem instances 20J20M2O, 40J20M2O, 60J20M2O and 80J20M2O respectively.

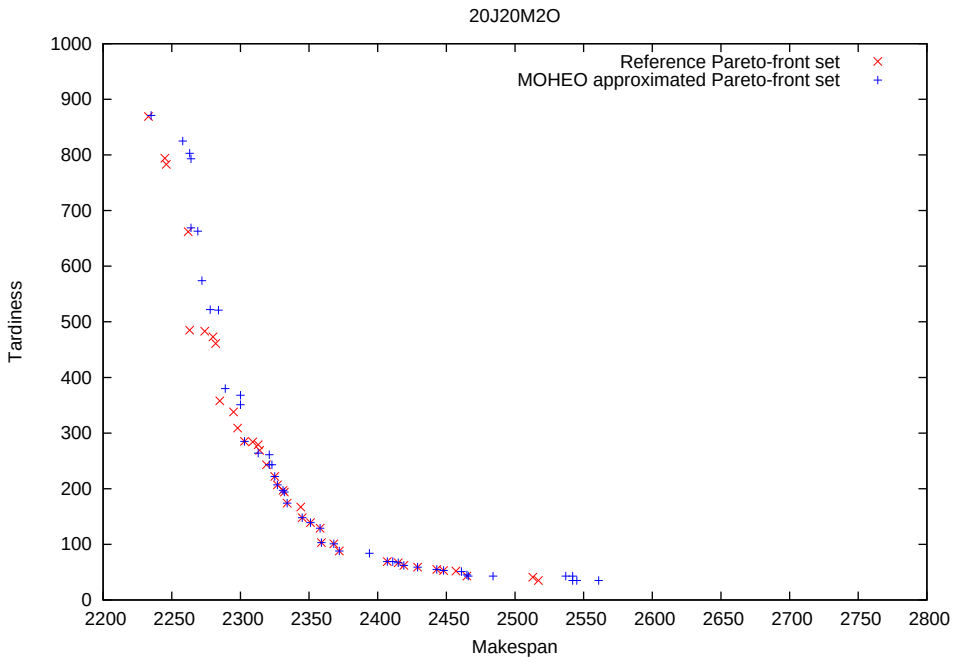


Figure 4.19: The MOHEO result of a single typical run for the bi-objective MOPFSSP test problems with 20 jobs and 20 machines versus the reference Pareto-front set.

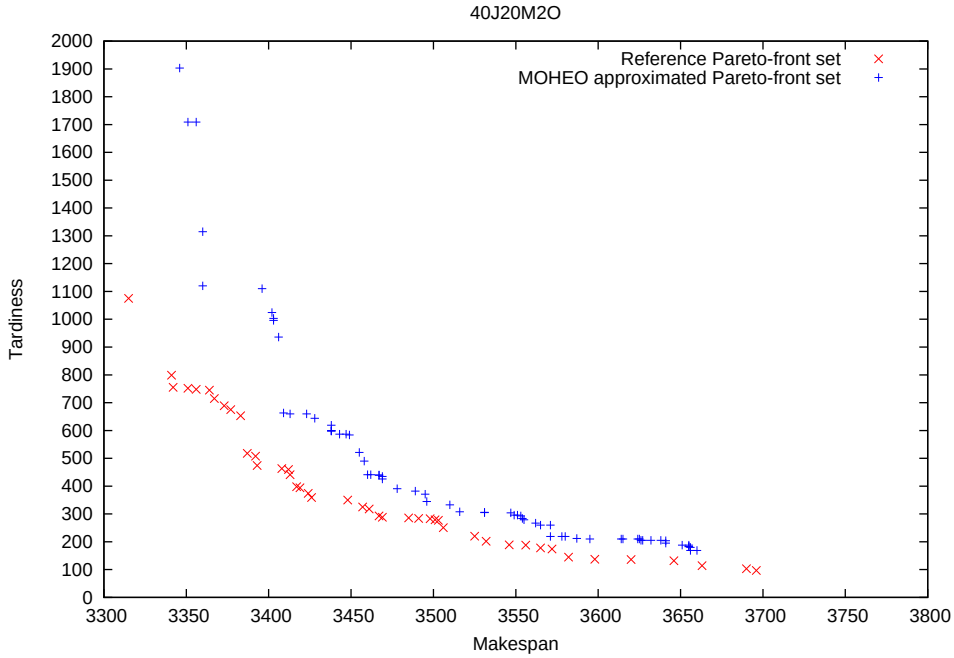


Figure 4.20: The MOHEO result of a single typical run for the bi-objective MOPFSSP test problems with 40 jobs and 20 machines versus the reference Pareto-front set.

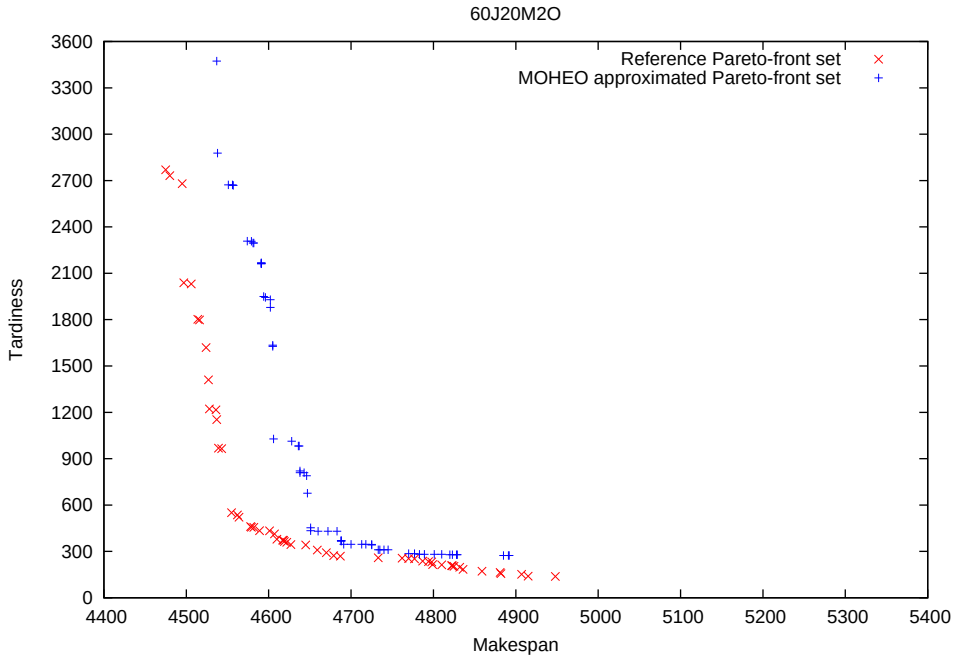


Figure 4.21: The MOHEO result of a single typical run for the bi-objective MOPFSSP test problems with 60 jobs and 20 machines versus the reference Pareto-front set.

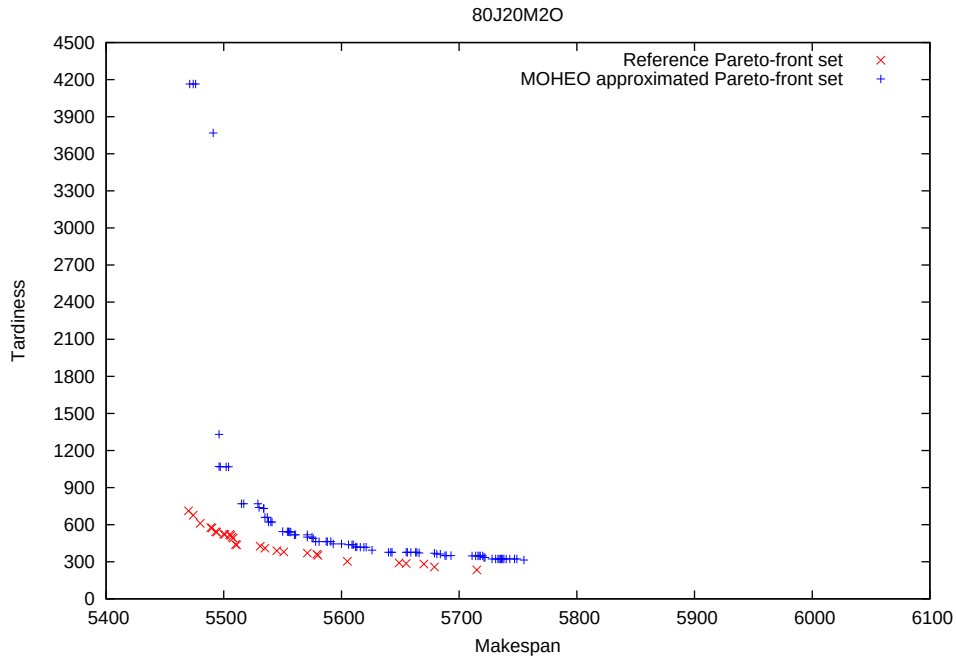


Figure 4.22: The MOHEO result of a single typical run for the bi-objective MOPFSSP test problems with 80 jobs and 20 machines versus the reference Pareto-front set.

Tables 4.13 and 4.14 present the S-metric and C-metric values with the results obtained for the four bi-objective permutation flow-shop scheduling instances 20J20M2O, 40J20M2O, 60J20M2O and 80J20M2O.

Problem name	20J20M2O		40J20M2O	
	$\bar{\chi}$	σ	$\bar{\chi}$	σ
MOHEO	383824.80	29818.37	633506.50	7965.27
Reference PS	400275.00	0	741811.00	0

Problem name	60J20M2O		80J20M2O	
	$\bar{\chi}$	σ	$\bar{\chi}$	σ
MOHEO	1160032.00	15756.96	1054911.00	25865.16
Reference PS	1409720.00	0	1173870.00	0

Table 4.13: The S-metric values for the 20J20M2O, 40J20M2O, 60J20M2O and 80J20M2O test problems. The values are the averages and the standard deviations for ten runs.

Problem name	20J20M2O		40J20M2O	
	MOHEO	Reference PS	MOHEO	Reference PS
MOHEO	-	0.010526	-	0
Reference PS	0.581896	-	1.000000	-

Problem name	60J20M2O		80J20M2O	
	MOHEO	Reference PS	MOHEO	Reference PS
MOHEO	-	0	-	0
Reference PS	1.000000	-	1.000000	-

Table 4.14: The C-metric values for the 20J20M2O, 40J20M2O, 60J20M2O and 80J20M2O test problems. The values are the averages for the ten runs.

4.4.3.2 Discussion

The MOPFSSP is a very interesting problem to be analysed because it has some features which are not conducive to the hybrid extremal optimisation framework.

The results for the small 20 jobs 20 machines instance show that MOHEO was able to find very close results with respect to the reference Pareto-front set. However, it is necessary to note that the results were more favourable with respect to the tardiness objective than the makespan objective (see Figure 4.19). The 20J20M2O instance presents an effective achievement in relation to the features of coverage, diversity and convergence. Although the approximate Pareto-front set that was found for this instance does not match the reference Pareto-front set, the former was very close to the latter. Through the S-metric value for this instance, it can be observed that the area in the search space that MOHEO covers is 4% less than the space covered by the reference Pareto-front set. Besides, the C-metric indicates that the solutions obtained by MOHEO dominate 1% of the solutions in the reference Pareto-front set but the latter only dominates 58% of the solutions obtained by MOHEO.

On the other hand, the results showed a different performance for the instances 40J20M2O, 60J20M2O and 80J20M2O. The three instances manifested a relatively efficient achievement with respect to the coverage and diversity; however the convergence demonstrated an evident difficulty in approaching the objective associate to the makespan (see Figure 4.20, Figure 4.21 and Figure 4.22). This inconvenience leads to a lack of capacity to achieve an appropriate convergence towards the reference Pareto-front set, which is also evidenced in the C-metric value. Here, none of the results obtained for the three instances were able to dominate any of the solutions in the reference Pareto-front set, which in turn was able to dominate all solutions for the results obtained by MOHEO (see Figure 4.14). Now, with respect to the S-metric it

can be observed that the area in the search space covered by these three instances is, on average, 14% less than the space covered by the reference Pareto-front set.

The reason for obtaining these results can be explained because of the impossibility of achieving an adequate assessment of the contribution for each component in the solution. This difficulty is triggered by the lack of information or data associated with each component, such as a value of cost, profit or weight. It is believed that these values should be integrated, rather than isolated values, into the objective functions or constraints.

The data that comes with each problem includes the number of jobs, the number of machines, the due date for each job and the processing time for each job in each machine. Here, the processing time is the more relevant value to assess the contribution for each component. However, this value becomes less relevant because all jobs are present in the solution. Thus, both objective functions, makespan and tardiness, must try to find a way to minimise the idle time gaps that could be generated in the machines, given the current permutation of jobs. In the case of the objective based on tardiness, the processing time has a more functional part as this value can be used to calculate the tardiness associate with each job in the current solution. This tardiness value can be used to define the component that degrades the solution. However, for the objective function based on makespan, it is not possible to calculate a value that can be associated to each job. This is because the makespan is always related to the last job in the permutation. Hence, this last job, which is represented by the last component of the solution, will always be designated as the component that degrades the solution. This means that the variability for the solutions in the makespan objective function will be less. For this reason the convergence associated with the makespan is deficient.

These types of problems, where the information necessary to carry out an appropriate fitness evaluation for each component of the solution is not satisfactory, become a new challenge to be taken up by the multi-objective hybrid extremal optimisation framework. Therefore, a more comprehensive study about the incorporation of an inseparable fitness evaluation technique for problems that do not provide the necessary information to accomplish a separable evaluation for each component, is required. This will extend the range of problems that the multi-objective hybrid extremal optimisation framework could solve.

4.5 Summary

This chapter has described the MOHEO framework to solve multi-objective constrained combinatorial optimisation problems. Firstly, the extension of the differentiated fitness evaluation scheme described in Section 3.2.1 was performed. This extension was developed by the incorporation of an aggregating function approach, which allows the assessment of the contribution for each component of the solution when there are two or more objective functions. Secondly, the extension of the double traverse local search as the secondary search mechanism described in Section 3.2.2 was carried out. This extension is elaborated through the incorporation of a nomadic double traverse system based on the modification and mixture of the weighted scalar fitness and the lexicographic ordering.

The integration of both extensions makes possible the implementation of the MOHEO framework. The former is used as the main search mechanism to carry out a coarse-grain exploration on the search space, taking advantage of particular characteristics such as its small requirements for parameterisation and implementation, and its stochastic nature of exploration. The latter is used as a secondary search mechanism to carry out a fine-grain exploration on the search space by refining new feasible solutions found in the main framework and improving the convergence toward the true Pareto-optimal set. Also it is able to expand the surface dominated by the Pareto-front towards their ends, discovering new non-dominated points.

The computational study has proved that the alliance of a simple extension of HEO and a secondary local search technique to handle multi-objective combinatorial problems, is competitive and efficient. However, when the information that comes with some problems is not enough to implement an adequate evaluation of the components of the solution, MOHEO has difficulties to converge towards some objective functions.

For the MOKP, MOHEO demonstrated a similar coverage when it was compared with the true Pareto-optimal set and a better coverage with respect to SPEA2 and NSGA-II, that is the MOKP experiments performed better for larger problems. The achieved diversification was acceptable for most instances.

The best results were obtained with MOQAP, having an almost perfect output. The experiments for MOPFSSP showed a drawback with regards to MOHEO when there was not enough information to evaluate the components' fitness. The results for the smaller instance with 20 jobs were reasonably competitive; however, the larger problems evidenced a deficiency in the convergence towards the objective associated with the makespan.

The results reported are preliminary, but promising. This study highlights the need to implement an inseparable fitness evaluation technique for problems that do not provide the necessary information to carry out a separable evaluation for each component, which will be addressed in the next chapter. Future work will also include a population-based extension of MOHEO to improve the performance of the approach. Furthermore, a new series of experiments will be executed to solve more complex test problems for MOKP, MOQAP and MOPFSSP. Also, there exists a considerable number of other multi-objective combinatorial problems that could be tested using MOHEO.

4.6 Contributions

Based on the theoretical development and the results of the experimentation phase, it is evident that the following objective has been achieved:

- To present an initial extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for multi-objective cases.

With the fulfilment of this objective, it is believed that the work developed in this chapter has contributed to the knowledge of constraint handling, extremal optimisation and hybrid methods for multi-objective CCOPs in the following ways:

- The multi-objective differentiated fitness evaluation scheme is developed as an extension of the single objective version for extremal optimisation. This extension allows extremal optimisation to solve single and multi-objective problems with or without constraints for the representation of solutions that generate either exclusively feasible solutions or infeasible solutions.

- The multi-objective double traverse local search algorithm is developed as an extension of the single objective version. This algorithm incorporates a modified weighted scalar fitness, which is merged with the lexicographic ordering method. As a result, a mechanism that allows the single solution in EO to travel towards the ends of the approximate Pareto-front set in a nomadic way, is achieved. That is, the single-solution permanently walks on the landscape looking for better non-dominated points.

- A multi-objective hybrid extremal optimisation framework is generated as an improvement to the hybrid extremal optimisation framework. This is the first

multi-objective version based on the canonical extremal optimisation meta-heuristic to solve multi-objective CCOPs. Thus, MOHEO responds to the need for implementing a multi-objective version of EO as was determined in Section 2.3.2.2.

- A number of derivate research areas as mentioned in Section 4.5 emerge from the work developed in this chapter. This offers the possibility to investigate the capacity of MOHEO to solve new multi-objective CCOPs, and thus improving the MOHEO mechanism, especially in regards to convergence, given the addition of new features.

Chapter 5

Application in RFID Antenna Design

5.1 Introduction

The genesis of Radio Frequency IDentification, or RFID, is associated with the creation of the radar system [181]. Radar was created with the idea of identifying objects by sending out a stream of radio waves which in turn bounced off objects, returning the signal to the radar, which could identify the existence of objects as well as their distance and speed. From these concepts, Harry Stockman published in 1948 the work, *Communication by Means of Reflected Power* [276], which is considered the seminal work on RFID. Nowadays, RFID is the technology used to identify, in an automatic way, objects at a distance without any physical or sight contact. This has lead to RFID being widely used in applications such as smart cards, electronic passports, access control, container identification, animal identification, sporting events, medical applications and industrial automation [114]. RFID requires a tag, which consists of a microchip together with a radio antenna. This tag is used to uniquely identify the bearer of it. One of the main challenges in creating this tag is the design of the antenna. Engineers must seek new ways to make an antenna as small as possible but still one with powerful transmission capabilities. With a small antenna, the tag becomes more tightly packed and is easier to transport by the bearer. However, the design of these antennas has been developed, not so long ago, through an empirical human process using the technical know-how gained in the area. This has mainly happened due to the lack of analytical methods which allow creating a design from any structure. Also, it implies a process that takes time and which generally does not achieve the most effective design possible.

Nature-inspired meta-heuristics, that have the ability to explore large and complex search spaces, looking for one or more optimal solutions, can be used as search mechanisms to try to find approximated optimal configurations in the design of RFID antennas. There was a first attempt in design, a meander line RFID antenna, using genetic algorithms [201]; however, this is an isolated work centred on simple and regular configurations, in which there has been no new research. A more systematic and general investigation has been developed using two other meta-heuristics to design meander line antennas. They are the Ant Colony Optimisation (ACO) [192, 193, 246, 300, 301] and Differential Evolution (DE) [216]. ACO has demonstrated that it is possible to apply a nature-inspired meta-heuristic to develop a software system to design and evaluate meander line RFID antennas. After a series of successful works with ACO there is a concern about whether it is possible to apply other nature-inspired meta-heuristics. Thus, a first work using DE has been developed. This research demonstrated that other meta-heuristics, like DE, can be used to obtain competitive results to those obtained with ACO for a multi-objective benchmark problem. Until now, no other meta-heuristics have been used to compare results. The application of new meta-heuristics would validate the use of nature-inspired methods to design RFID antennas and also to verify that the results which have been found so far for the multi-objective benchmark problems, are the best that can be found.

The above has motivated the development of an extremal optimisation application using the MOHEO framework, to be employed in the design of RFID antennas. The implementation of this approach has an interesting challenge because the evaluation for any valid solution of an antenna layout is performed by an external black-box module provided for the free antenna modelling software package, NEC [49]. Hence, this situation implies that there will not be the information needed to make an assessment at the level of the components of the solution, thus requiring an inseparable fitness evaluation. Herein, is presented a multi-objective hybrid extremal optimisation approach to solve RFID antenna designs, using an inseparable fitness evaluation technique. The inseparable fitness evaluation technique will be made by borrowing the idea of the global pheromone scheme used in ACO. This scheme will serve to maintain a record, over generations, of the components present in the solutions that have a good evaluation. Thus, components with a higher presence in good solutions will have a higher level of pheromone, which is interpreted as that these are components that contribute to a good quality solution.

This novel inseparable fitness evaluation technique will be incorporated into the MOHEO framework presented in Chapter 4. This extension will allow MOHEO to

solve a new range of problems, those that have not enough information to perform a separable fitness evaluation. For example, many industrial processes work with Programmable Logic Controllers (PLCs) [299], which are used for the automation and control of machinery in a factory. The optimisation of these industrial processes, through the application of a nature-inspired meta-heuristic based on extremal optimisation, could require detailed information about the data delivered, by the PLC devices, in order to perform the fitness evaluation for the components of the solution. However, usually this additional information is not available due to the fact that PLCs work in terms of input and output data, without any knowledge of the internal working. For these cases, the inseparable fitness evaluation technique is a useful extension for meta-heuristics based on extremal optimisation, in order to solve problems which have an inseparable fitness evaluation feature.

The MOHEO approach is tested with the same set of benchmark problem instances used by Lewis et al. [193] for the multi-objective ACO approach and Montgomery et al. [216] for the multi-objective differential evolution approach. Thus, the results obtained with MOHEO are compared against those obtained by the two works previously indicated.

This chapter pursues the following objectives:

- To present an inseparable fitness evaluation technique for the extremal optimisation heuristic to handle problems that do not provide the necessary information to perform a separable evaluation of components.

- To present an initial extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for a real-world multi-objective case.

The rest of this chapter is organised as follows. Section 5.2 gives a brief description of meander line RFID antennas and the nature-inspired methods applied to solve it so far. Section 5.3 explains an inseparable fitness evaluation scheme to solve problems that do not provide the necessary information to perform a separable evaluation of components. Section 5.4 presents how the multi-objective hybrid extremal optimisation is applied to the design of meander line RFID antennas. Section 5.5 gives a summary of the computational experiments developed with an analysis of them. Section 5.6 gives a summary of the chapter and discusses the future work arising from this study. Finally, Section 5.7 states the contributions that arise from this chapter.

5.2 RFID Antennas

RFID systems are comprised of a set of elements that together allow it to carry out an identification process. The most important components are the RFID tag, which consists of two basic components: a chip that contains the identification code, with a small memory in some cases, and the antenna that receives the requirement from the RFID reader and returns the identification code. The RFID reader is responsible for sending a radio frequency signal that can power the RFID tag through the energy it harvests via the radio waves from the reader. With this collected energy, the tag activates the circuit and transmits radio waves back to the reader with the identification code. The antenna needs to be highly efficient to harvest sufficient energy to subsequently re-radiate with sufficient efficiency so that the reader within the read range can reliably receive the return signal. The read range is the coverage area where the RFID tag is able to receive the signal from the RFID reader with enough power to energise the tag, activate the chip and return the identification code. Finally, there is a host computer that obtains the identification codes captured by the RFID reader for further processing, activating certain mechanisms or actions. Figure 5.1 gives a schematic of the different components which make up a simple RFID system.

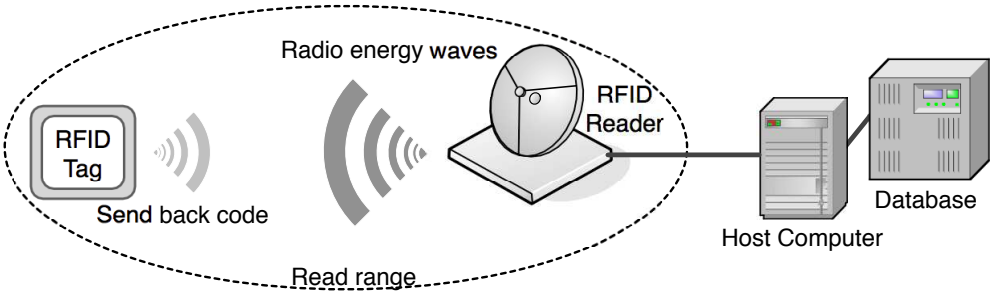


Figure 5.1: The simple RFID system.

The optimisation of the design of RFID antennas has the objective of maximising the read range using an antenna as small as possible. To accomplish this, technically speaking, it is necessary to maximise the antenna efficiency, or gain, and minimise the resonant frequency. The efficiency or gain, for an antenna, is the state of maximum radiation that can be achieved for the transmission. Thus, the read range is directly proportional to the efficiency. On the other hand, the energy required to transmit a

radio wave is directly proportional to its frequency, i.e. low frequency signals require less energy. The resonant frequency is the frequency at which a given antenna transmits the maximum energy as a signal (instead of dissipating it, typically as heat). So in an RFID antenna, with a very small amount of energy available for the return signal, the read range is increased if the frequency is lower. That is, more energy can be used to make the signal greater in amplitude or strength. RFID systems operate at particular standard frequencies and researchers try to get the best match of the antenna resonant frequency with this operating frequency. Smaller antennas (desirable for RFID) naturally have higher resonant frequencies (undesirable for RFID). Hence, the lower the frequency, the greater the antenna size [201]. For this reason, the research work is addressed to find small antenna designs with low resonant frequencies and high efficiency.

Meander line RFID antennas are folded dipole antennas that can be represented by a Cartesian grid [121, 246], which is symmetrical around the dipole. Hence, for the purposes of the optimisation process, only one half of the antenna will be used to design it. The grid is comprised of a finite number of points in a square shape. The meander line is created by the connection of these points by horizontal or vertical segments, so that no segment intersects another already established. Generally, all grid points should be connected, however this is not a mandatory restriction. One constraint is given by the fact that the meander line must start on any boundary point on the edge of the grid that is parallel and closest to the symmetric line that divides the two halves of the antenna. From there, a continuous line through the points of the grid is formed. The start point is used to connect, through an additional segment, the two-mirrored grids that form the dipole antenna. Figure 5.2 shows, using three illustrations the Cartesian grid for a 5×5 antenna.

The creation of a meander line for the design of RFID antennas is related to two widely applied problems, which are the Traveling Salesman Problem (TSP) [248] and Self-Avoiding Walks (SAWs) [231, 267]. The objective in the TSP is to find a path that connects all points in the grid so that each point is visited once and finally return to the starting point. Note that in the TSP, any point can be connected to any other point. In the case of RFID, a path that connects all points must also be found, but the rules to connect the points are more restricted than the TSP. For instance, for an RFID antenna, each point can only be connected to the nearest points in a vertical or horizontal way (i.e. non-diagonal); that is, those directly adjacent. Also, for an RFID antenna, it is not necessary to return to the initial point, as is the case of the TSP. This means that RFID is a restricted version of TSP. On the other hand, the

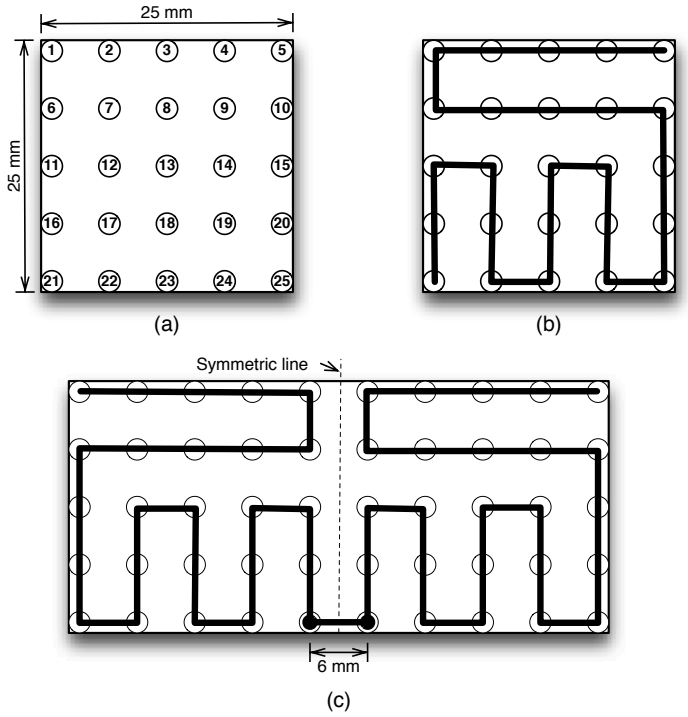


Figure 5.2: (a) The grid defines a 5×5 antenna. (b) Illustrates a potential meander line antenna for the grid defined in (a). (c) The dipole antenna generated through the binding of two mirrored meander line antennas shown in (b) by a 6 mm join.

aim in SAW is to find a path that connects the grid points without generating any crossing points on it. This closely resembles the creation of a meander line RFID antenna; however, the main difference is in the dimension of the grid defined for both problems. In the RFID case, the dimension of the grid is given by a finite number, which is generally small in size. Contrarily, in the SAW case, the dimension of the grid is infinite due to the fact that this problem is mainly used in chemistry for simulation of linear polymer molecules in a good solvent, and polymers exist in continuous space. The knowledge of these two other problems allows for a better understanding of the modelling of RFID antennas as a combinatorial problem, and subsequent application of nature-inspired meta-heuristics.

Recently, meander line RFID antenna design has been successfully solved by an intelligent automated process using nature-inspired meta-heuristics such as ACO and DE. Randall et al. [246] developed an initial software system to design and evaluate meander line RFID antennas based on ant colony optimisation. The approach was

successfully applied to solve the design of meander line RFID antenna on grids from 5×5 to 10×10. The ACO implementation works as a constructive mechanism creating a Hamiltonian path on the grid. The NEC evaluation suite performed the assessment of potential solutions. As NEC requires a considerable length of time to evaluate a solution, a cache mechanism was implemented to store the evaluations found for previous solutions. The antenna efficiency was considered the objective to be maximised. For the 5×5 grid, for which the optimal result is known [121], the approach was able to obtain an antenna design very near the optimal. For the remaining instances, in general, very efficient antenna designs were found.

In an effort to improve the results of this preliminary research, Weis et al. [300] proposed an extension based on a local search technique that uses an operator known as backbite. This operator is commonly used by the SAW problem for the construction of compact polymer chains. The backbite operator generates a new meander line antenna from another meander line antenna by the addition of a new segment, within the legal scope of alternatives, at the end point. The incorporation of the new segment generates a circuit, which is solved by the elimination of other segments in the area of the grid where the mismatch was generated. Results show an improvement in the efficiency for all grid instances, from 5×5 to 10×10, compared to those obtained in the previous work by Randall et al. [246], especially for larger grid sizes.

Until now, the optimisation in the RFID antenna design has only considered the efficiency maximisation. However, the minimisation of the resonant frequency is also an important aspect that should be considered. Thus, Lewis et al. [192, 193] proposed a multi-objective version based on the two previous works. In this investigation the design of both high and low frequency antennas was examined. The incorporation of the new objective allows the elimination of the constraint regarding that the meander line, that it must cover all points on the grid (that is, be a Hamiltonian walk). Results obtained by the approximated Pareto-front sets, for the range of grid instances from 5×5 to 10×10, demonstrated that it is possible to produce highly efficient antennas with low resonant frequencies and with a shorter runtime for the high frequency cases than the low frequency cases.

A new work developed by Weis et al. [301] is appended to the previous three where a mechanism is proposed to increment the performance of the software system in order to design and evaluate RFID antennas based on ACO. The new proposal reflects the a-priori knowledge and practical experience from makers and designers regarding good or acceptable solutions. These solutions are incorporated into a pre-seeding procedure, which generates an initial pheromone biased by those solutions.

The new enhancement obtained very good solutions and also allowed a fine tuning of the search procedure to occur. An achievement of this study was the decrease in the time to find efficient designs, which led to the possibility to extend it and solve more complex levels of optimisation problems, such as tri-objective problems.

The latest work used a new nature-inspired meta-heuristic, different to ACO, to implement the software system in order to design and evaluate RFID antennas. Montgomery et al. [216] developed the first multi-objective differential evolution approach, whose results could be compared with those generated by the pioneering ACO version. The main dissimilarity between ACO and DE is the way the solution is created. The former use a constructive mechanism where each solution is gradually constructed by incorporating the various segments that make up the meander-line in the antenna design. In contrast, the latter starts with a complete initial solution, which is iteratively improved using vector differences to modify it. The results showed that DE is an appropriate method to achieve good solutions. The differential evolution approach was more effective for smaller antenna grid sizes, decreasing its performance as the size increased. The latter may occur because the DE implementation did not use any local search mechanism to improve the solutions found by it.

The work performed with DE evidently demonstrates that it is possible to implement non-constructive meta-heuristics to solve the RFID antenna design problems, which is inherently constructive in nature. However, differential evolution is a population-based search method that operates in continuous domains and it has been commonly used to solve continuous optimisation problems. Thus, considering that the RFID antenna design problem has been modelled as a combinatorial problem, it could be expected that other non-constructive meta-heuristics could also achieve competitive results to those generated by the ACO approach. Indeed, it is believed that extremal optimisation, whose search method is especially suitable for solving combinatorial optimisation problems, is a good candidate to achieve this purpose.

The next sections focus on the development of an extremal optimisation version to implement the software system to design and evaluate RFID antennas.

5.3 Inseparable Fitness Evaluation Scheme

One of the main features of extremal optimisation that makes it different to other meta-heuristics, such as evolutionary algorithms, is its component fitness evaluation scheme. Here, each component of the solution is evaluated to assess its contribution in obtaining the optimal solution. To do this, it is essential that the problem to be solved

has as much suitable data or information as possible for this purpose. However, not all problems can provide this information. For this reason, it is necessary to provide some other mechanism that can be incorporated into extremal optimisation in order to address these problems, which are not possible to be carried out as separable component fitness evaluations.

In the case of the RFID antenna design problem, the available input data has a relationship to the physical features of the antenna, such as the grid size used to design the antenna, the side antenna size, the width of the wire and the length of the wire segment that connects the dipole. This information, along with a proposed antenna design, is used as input to the NEC analysis software. This software works as a black box computation, which returns the evaluation with regard to both the efficiency and resonant frequency for the proposed antenna design given as input. No other data is available or known about these calculations, which may be used to perform the component fitness evaluation.

Herein, an inseparable fitness evaluation based on the pheromone structures used by ant colony optimisation is proposed. As it was described in Section 2.2.3, the level of pheromone is updated in two stages. One possible scheme is as follows and is based on Ant Colony System (ACS) [104]. Firstly, each time an ant traverses a path from the nest to the food source, a local pheromone update is done. This action is performed until a complete route is produced, updating the pheromone level at each section of the route. Once all the ants have finished the creation of a route from the nest to the food source, a second update is carried out. Here, the best route found at the current iteration is used as a global pheromone update for each section in this route. As extremal optimisation is not a meta-heuristic that generates a solution applying a constructive mechanism, only the global pheromone update is considered.

The idea behind the use of the pheromone structure lies in that this works as a element which can keep a record of the solution components that are present in good solutions. In this way, each time either a new best solution (for single-objective optimisation) or a new non-dominated solution (for multi-objective optimisation) is found, then a pheromone update is performed. More specifically, with this procedure the components that degrade the solution will have low pheromone levels and the component that contributes to the optimality of the solution will have a high pheromone level. Thus, the pheromone structure allows the implementation of a fitness evaluation of the components of the solution without any specific information about the evaluation of the objective function(s). Figure 5.3 shows the pheromone representation for the RFID antenna design problem.

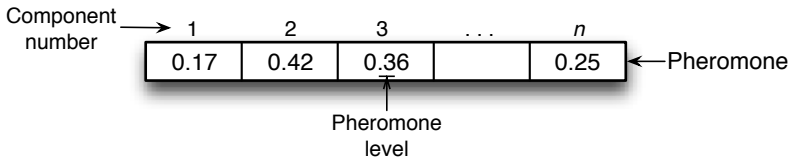


Figure 5.3: An example of a pheromone vector for the RFID antenna design problem, where n is the number of components in the solution.

The pheromone vector is initialised with a very low value. After that, the pheromone level for each component is updated along an extremal optimisation process according to Equation 5.1.

$$\Phi_i^t = (1 - \alpha)\Phi_i^{t-1} + \alpha\Delta \tag{5.1}$$

where:

- Φ_i^t is the pheromone value for the i^{th} component at time t,
- α is the pheromone decay factor, and
- Δ is the pheromone positive reinforcement for the i^{th} component if it belongs to a non-dominated solution, 0 otherwise.

Finally, the inseparable fitness evaluation scheme is given by Equation 5.2.

$$\lambda(x_i) = \Phi(x_i), \quad \forall i \quad 1 \leq i \leq n \tag{5.2}$$

where:

- $\lambda(x_i)$ is the fitness evaluation for the i^{th} component of the solution,
- $\Phi(x_i)$ is the pheromone level for the i^{th} component of the solution,
- n is the number of components of the solution.

The next section explains how the multi-objective hybrid extremal optimisation framework with an inseparable fitness evaluation scheme is applied to solve RFID antenna design problems.

5.4 MOHEO Applied to RFID Antenna Design

The implementation of the multi-objective hybrid extremal optimisation framework to solve the RFID antenna design problem uses as its basis, the previously developed work in the area presented in Section 5.2.

Until now, all developed approaches with either ant colony optimisation or differential evolution, have worked with the meander line antenna design. Firstly, the

works were focussed on creating a Hamiltonian meander line; that is, all the nodes on the grid must be connected by a segment without generating a closed circuit. After that, the requirement of generating a Hamiltonian walk on the grid was relaxed, which allowed the generation of meander line antennas with different lengths. The previous requirement was imposed for the single-objective approach where the objective function only measures the efficiency of the antenna and the resonant frequency was minimised generating the longest antenna possible. However, with the bi-objective approach, where the resonant frequency is addressed as an objective, the Hamiltonian walk requirement was not necessary anymore.

Previous works fulfilled the objective of demonstrating that it is possible to develop a software system to design and evaluate meander line RFID antennas using both a constructive meta-heuristic, like ant colony optimisation, and a non-constructive meta-heuristic, like differential evolution. Besides, two exploratory studies were conducted. The former worked with a guided search using a pre-seeding pheromone, in the case of the ant colony optimisation approach. The latter investigated the variant of building the antenna using a wire with different widths. All this has improved both the results found and the problem solving process. However, new questions emerge as to the possibility of applying other nature-inspired meta-heuristics for generating antenna designs and if it is possible to build efficient antennas with a different design to that of a meander line.

In response to these questions, the idea of developing a MOHEO approach applied to the RFID antenna design using modified meander lines, is proposed. The idea of working with a modified meander line emerges from the natural fit that extremal optimisation has to generate this type of antenna. In this way, the limitations and extra computational work that extremal optimisation would require to generate meander lines, are eliminated. Thus, the restriction of generating a single continuous line is relaxed and now a line with loops, mesh segments and parasitic elements can be designed. This revised scenario involves a new interpretation of how the grid is built.

In previous works, each *node* on the grid is tagged with an identification number from 1 to n^2, where n is the grid size, as was shown in Figure 5.2. Therefore, the solution is represented by a sequence of nodes, which are joined in order by a continuous line to generate the antenna design. For example, in Figure 5.2 the solution S for the grid is $S = \{21, 16, 11, 12, 17, 22, 23, 18, 13, 14, 19, 24, 25, 20, 15, 10, 9, 8, 7, 6, 1, 2, 3, 4, 5\}$, which starts from node 21. However, in a modified meander line antenna design scheme, the same grid has a different interpretation. Here, the *segments* or *edges* that join the nodes are tagged with an identification number from 1 to $(2 \times (n-1) \times n)$,

where n is the grid size, as is shown in Figure 5.4.

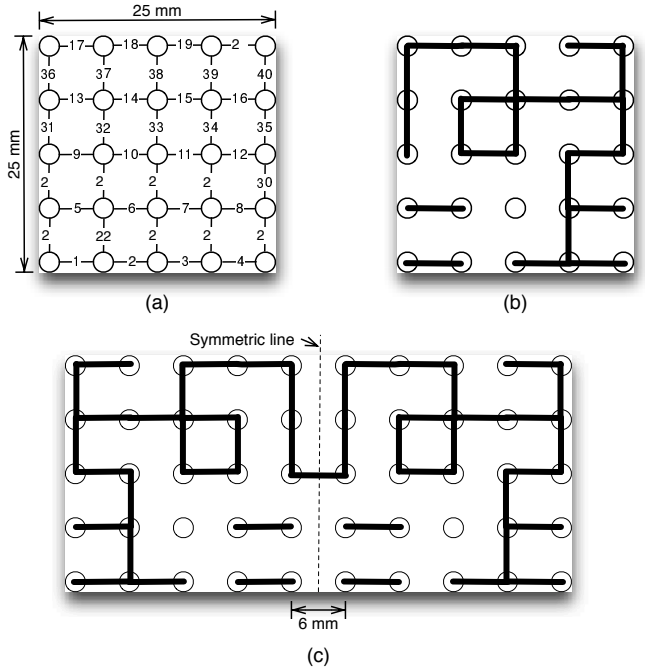


Figure 5.4: (a) The grid representation for a 5×5 antenna based on segment allocation. (b) Illustrates a potential modified meander line. (c) The dipole antenna generated with two mirrored modified meander lines connected by a 6 mm segment.

For the modified meander line case, the solution is represented by a vector of $m = (2 \times n \times (n-1))$ components. Each component represents a segment that could potentially be put in the antenna design. If the i^{th} segment is allocated to the antenna design, then a value of 1 is assigned to its position in the solution vector. On the other hand, a value of 0 is assigned to represent the segment is not allocated to the antenna design. Figure 5.5 shows a representation for the modified meander line antenna design problem.

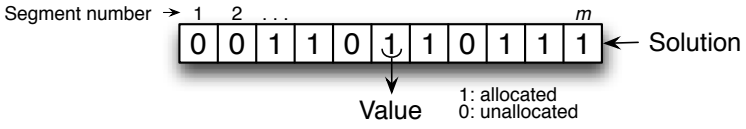


Figure 5.5: An example of a solution vector for the modified meander line antenna.

Thus, for the antenna design given in Figure 5.2 for a grid of size 5×5, the solution vector S, in a modified meander line antenna design, is made up by the assignment of the value 1 in the positions 2, 4, 9, 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 35 and 36 with all remaining positions having the value 0, as is shown in Figure 5.6.

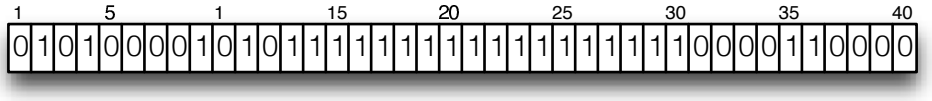


Figure 5.6: The modified meander line solution that is equivalent to the meander line solution for the example given in Figure 5.2.

This representation to design a general modified meander line antenna, becomes a combinatorial optimisation problem similar to the knapsack problem. The main difference between this and the problem defined in this chapter, is that the weight for each knapsack item is known, whereas in the modified meander line antenna design there is no information related to any sort of weight that is associated with the segments. Also, there is the constraint that a segment associated to the edge of the grid aligned to the symmetric line, must be allocated. Taking into consideration these differences, herein, the modified meander line antenna design will be formulated as a multi-objective knapsack problem.

The initial solution is obtained by a random selection of segments. This mechanism could generate an infeasible solution due to the requirement that at least one segment associated to the edge of the grid aligned to the symmetric line, must be in the antenna design. This is because of the additional 6 mm segment that is used to connect and energise the two-mirrored grids that form the dipole antenna. If an initial feasible solution is generated then this is added as the first point in the approximated Pareto-front set.

For this particular problem, which has no available information to perform an assessment of the components of the solution, the differentiated fitness evaluation is implemented with the support of the inseparable fitness evaluation scheme described in Section 5.3. Thus, Equation 5.3 illustrates the fitness evaluation to select the segment that degrades the solution to be assigned or unassigned from the antenna design.

$$\lambda(x_i) = \begin{cases} \Phi(x_i) & ,\forall i \quad 1 \leq i \leq m \quad \text{for feasible solutions} \\ \\ -V_i & ,\forall i \in \tilde{M} \qquad\quad \text{for infeasible solutions} \end{cases} \tag{5.3}$$

where:

$\Phi(x_i)$ is the pheromone level for the i^{th} component of the solution,
m is the number of components of the solution,

V_i is the number of segments connected to the i^{th} unassigned segment associated to the edge of the grid aligned to the symmetric line, and

$\tilde{M}$ is the set of segments associated to the edge of the grid aligned to the symmetric line.

When the solution is feasible, the inseparable fitness evaluation based on the pheromone structure is applied to define which components of the solution degrade the solution. Thus, segments that have a low level of pheromone are chosen with a higher probability to be allocated/unallocated than those with a high level of pheromone. This is because a segment with a low level means that it does not contribute to achieving a high efficiency with a low resonant frequency in the modified meander line antenna design. On the other hand, when the solution is infeasible, this means that none of the segments on the symmetric line is allocated. To obtain a feasible solution again, one of them must be selected to be part of the antenna design. Hence, segments on the symmetric line with a high number of neighbouring segments connected to it, are considered to be the components that degrade the solution. This is because a segment that is connected to other adjacent segments has a higher probability of generating a longer continuous line. This is important due to longer continuous lines reducing resonant frequency.

The Roulette Wheel Selection (RWS) is used to select one of the worst evaluated segments according to its probability P. The replacement of the value for the chosen component (segment) is a simple process because it only has to exchange its value from 0 to 1 or from 1 to 0, as appropriate.

Due to the very time-consuming process to evaluate the solution by the NEC software, for now, the secondary search is not performed. However, a local search performed manually, to discern a possible future implementation of this, is carried out at this stage of the computational experimentation. For the same reason, as in previous works, a caching system is implemented to reduce the number of evaluations made by the NEC software and so to reduce the runtime of the MOHEO process.

Each time a feasible solution is found, the non-dominated procedure is activated to update the approximated Pareto-front set found at the current iteration. If a new non-dominated solution is found, then the pheromone level update is performed. The MOHEO procedure is repeated for a pre-set number of iterations. Finally, the approximated Pareto-optimal and Pareto-front sets that were found by the approach are returned as output. Algorithm 23 shows the MOHEO pseudocode for the bi-objective modified meander line antenna design problem.

Algorithm 23 The MOHEO framework for the bi-objective modified meander line antenna design problem.

Generate the probability vector P
Generate the initial pheromone structure Φ
Initialise a solution S with random values from $\{0, 1\}$
if S feasible **then**
  Add S into the approximated Pareto-front set
  Update pheromone levels Φ
end if
repeat
  if the current solution is feasible **then**
    Evaluate the fitness $\lambda(x_i)$ for the components of the feasible solution
  else
    Evaluate the fitness $\lambda(x_i)$ for the components of the infeasible solution
  end if
  Rank the components according to its fitness $\lambda(x_i)$ from the worst to the best
  Select a component based on the probability of its rank P using RWS
  Obtain a S_{new} in the neighbourhood of S changing the value of the selected component to 0 or 1 as appropriate
  Evaluate the new solution S_{new}
  if S_{new} is feasible **then**
    Apply the $NonDominance(S_{new})$ procedure
    if a new non-dominated solution is generated **then**
      Update pheromone levels Φ
    end if
  end if
until the terminal condition is satisfied
return the approximated Pareto-optimal and Pareto-front sets

5.5 Computational Experiments

The proposed multi-objective hybrid extremal optimisation algorithm was coded in the C language and compiled with `gcc`. The NEC evaluation software used was the version available in late October 2010. The computing platform used to perform the tests had a 3.00 GHz Intel Core2 Duo CPU, 3.8 GB of memory and ran under a Linux 32-bit OS.

The test data for the RFID antenna design problem was the same as used in the previous works by Lewis et al. [192] for the ACO approach and Montgomery et al. [216] for the DE approach. These tests consist of a set of different grid sizes from 5×5 to 10×10. All grids have a dimension of 25×25 mm and the width of the wire used to generate the segment line is 1 mm. Also, there is an extra wire segment of 6 mm in length, which is used to connect the dipole in the symmetric line.

The number of iterations to complete the MOHEO search process was set to 1000.

For the grids of size 5×5, 6×6 and 7×7 an additional run with 2000 and 3000 iterations was performed to observe if lower resonant frequencies could be reached. Each instance was run three times and for each run the random seed was changed to a new value. At the moment, a higher number of runs were not performed because the solution evaluation of the NEC software takes a considerable amount of runtime. For example, one evaluation for a grid size of 5×5, can take between 15 to 90 seconds. As a consequence, for the same grid size instance, a search process with 1000 iterations may require, on average, about 9 hours and 30 minutes. These times become significantly larger as the grid size is increased.

The τ parameter was set at 1.4 for all instances. This value has been reported in previous research [42, 143, 243] as being a good value to obtain efficient solutions. The initial pheromone level and the pheromone decay facto were set at 0.1, as these values have been found to be robust by Dorigo and Gambardella [104]. The pheromone positive reinforcement was set at 1.0 as a constant value. Generally, this value is calculated as $1/L$, where L is the length of the best path generated by the ants. But, as extremal optimisation technique does not generate a path, the pheromone positive reinforcement was implemented as a constant value.

5.5.1 MOHEO Results and Analysis

Here, an analysis of the results obtained by the MOHEO approach is performed. The aim is to observe if MOHEO is able to generate an approximated Pareto-front set for the objectives of maximising the efficiency and minimising the resonant frequency in the RFID antenna design.

First, the considerable runtime required by the NEC antenna evaluation software is an issue that must be kept in mind. As previously mentioned, for an antenna design in a grid size of 5×5, the NEC software takes a runtime in the range of 15 to 90 seconds per evaluation. The runtime difference depends on the particular antenna design that requires evaluation. Thus, designs that energise the longest antenna segment lines require the shortest runtime in the NEC software. This runtime rises considerably as the size of the grid is increased. Thus, in a grid size of 10×10, the evaluation for an antenna design with the NEC software can reach a runtime of 19 minutes, on average.

Previous work on RFID antenna design leant towards the creation of antennas through a continuous line that follows the pattern of a meander or plough path. These continuous line antennas, besides obtaining a quick evaluation, also are able to generate designs with lower resonant frequencies as the longer the continuous line, the lower the resonant frequency. However, the MOHEO implementation works from a

different point of view, with the objective of investigating new antenna design patterns that achieve higher efficiency with lower resonant frequencies. For this reason, the approach proposed in this chapter works by simulating a knapsack problem through the incorporation or not of segment lines in the grid. This free allocation of segments in the grid allows for the generation of antenna designs that do not necessarily follow any predetermined pattern. Therefore, the probability of creating a continuous long line is lower. This feature has a strong impact on both the increasing evaluation runtime for the NEC software and the decreasing generation of antennas with lower resonant frequencies.

In order to corroborate this, a set of five runs for the three grids with lower size (5×5, 6×6 and 7×7) was performed. First, for each of them, three searches with 1000 iterations each were carried out by varying the initial seed used for the probabilistic operations of the algorithms. This was done to observe if MOHEO is able to generate similar approximated Pareto-front sets. After that, two new searches were performed with 2000 and 3000 iterations respectively. This has the aim of seeing if the approximated Pareto-front set converges towards lower resonant frequencies when the MOHEO approach has more search time. Figures 5.7, 5.8, 5.9, 5.10, 5.11 and 5.12 show the plotted graphs with the results of these computational tests.

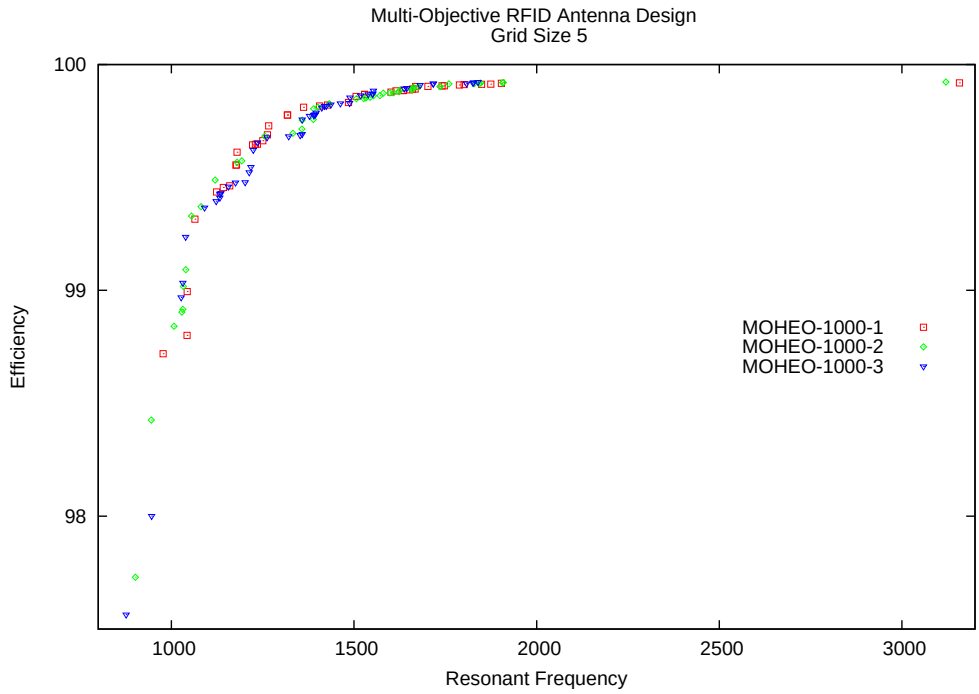


Figure 5.7: The MOHEO result of three single typical runs, for the 5×5 bi-objectives RFID antenna design problems, when the seed value changes.

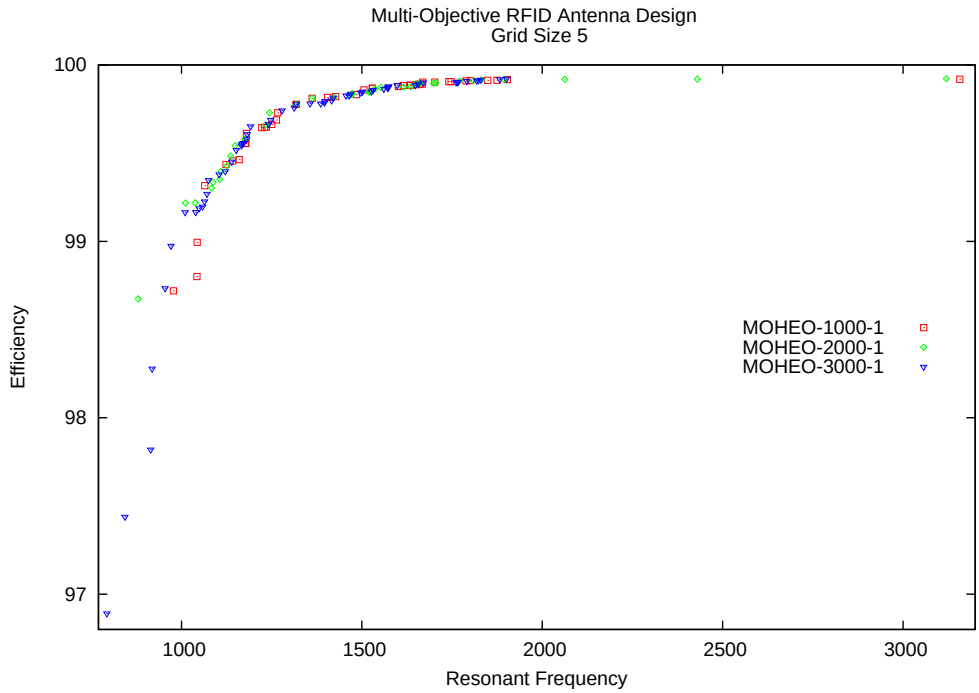


Figure 5.8: The MOHEO result of three single typical runs, for the 5×5 bi-objectives RFID antenna design problems, when the iteration value changes.

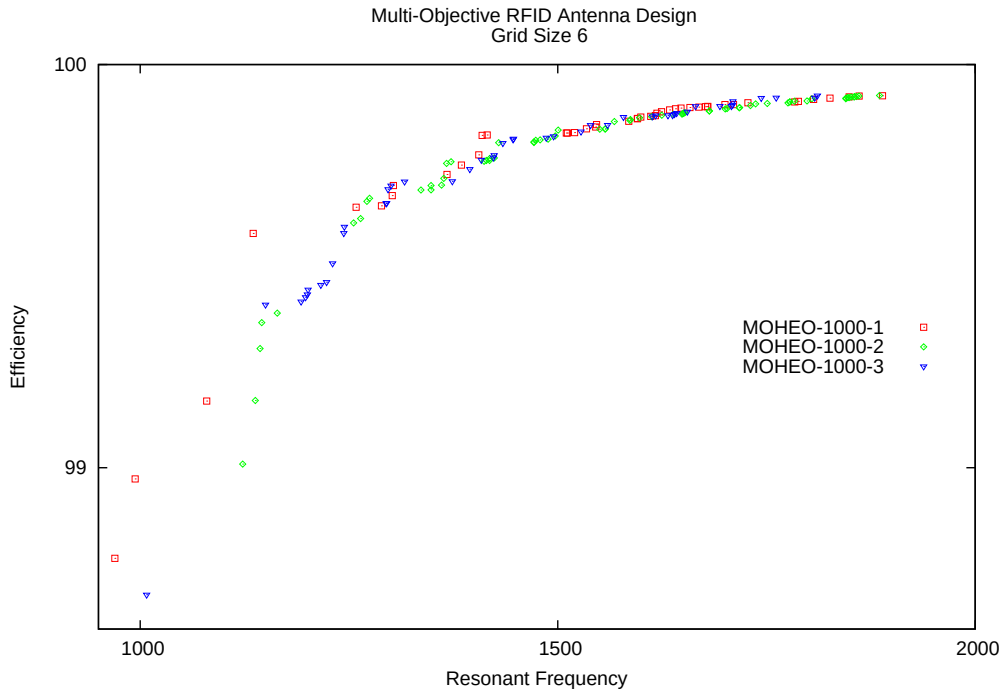


Figure 5.9: The MOHEO result of three single typical runs, for the 6×6 bi-objectives RFID antenna design problems, when the seed value changes.

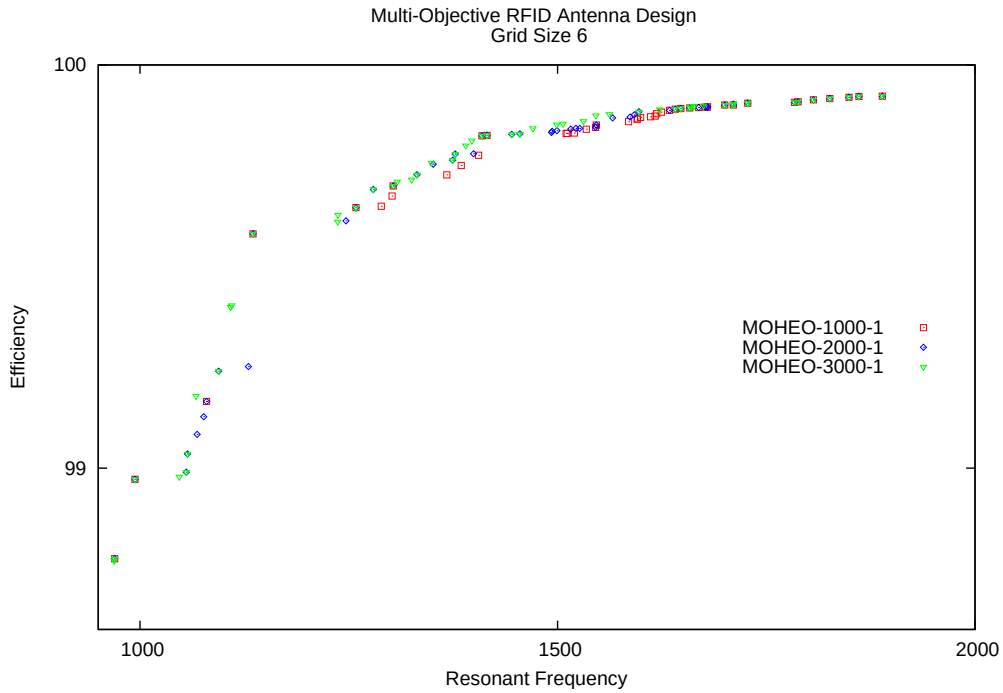


Figure 5.10: The MOHEO result of three single typical runs, for the 6×6 bi-objectives RFID antenna design problems, when the iteration value changes.

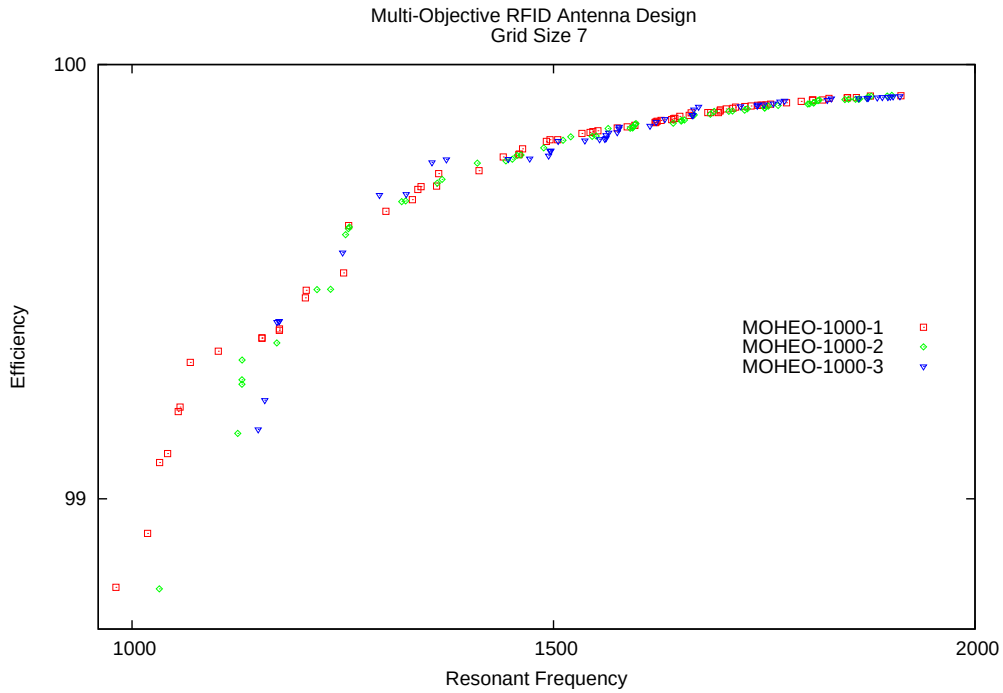


Figure 5.11: The MOHEO result of three single typical runs, for the 7×7 bi-objectives RFID antenna design problems, when the seed value changes.

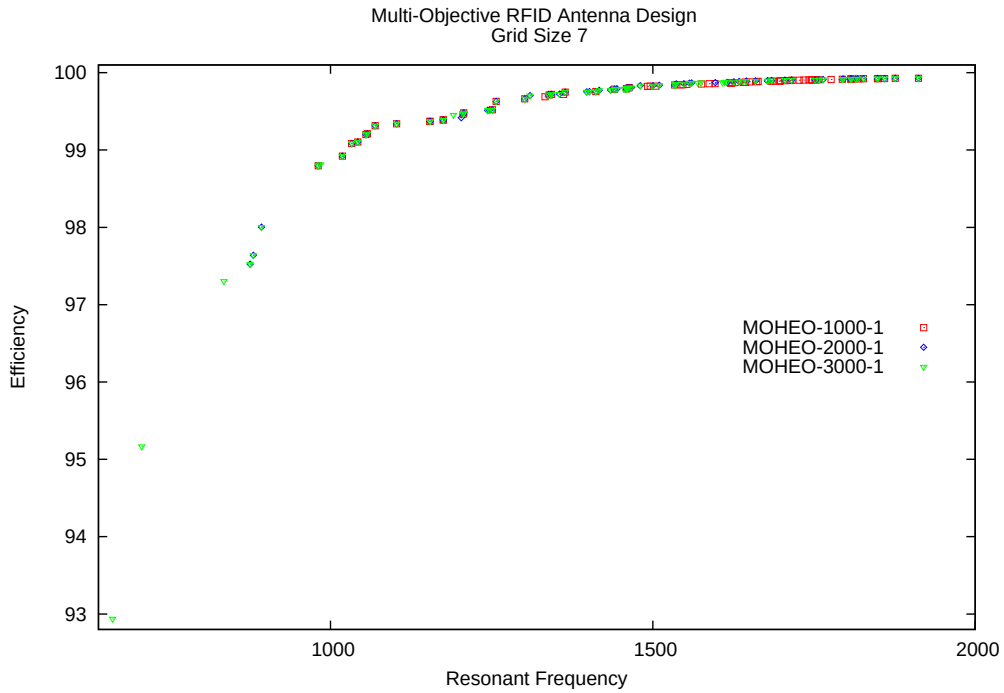


Figure 5.12: The MOHEO result of three single typical runs, for the 7×7 bi-objectives RFID antenna design problems, when the iteration value changes.

From the graphs, it can be seen that MOHEO is able to generate similar approximated Pareto-front sets when the initial seed varies for different searches with the same number of iterations. This indicates that the approach can work in a regular form for other searches. Also, when the number of iterations is increased, the approach is able to find a greater number of points with better convergence and scope. However, this improvement was not expected to reach solutions with lower resonant frequencies.

In an effort to see how the convergence towards lower resonant frequencies can be improved, a potential local search mechanism was analysed. This mechanism consists of trying to generate, in the current solution, a continuous line as long as possible through the modification of one or more segments. The difficulty of performing this simple operation lies in the generated antenna design by MOHEO not following any predefined pattern such as a meander line (see Figure 5.4). The drawn line on the grid by the MOHEO can simultaneously produce continuous lines, branched lines, circuits and completely isolated lines. A heuristic that can be followed is to identify all end segment lines and to try to add a new segment without creating a circuit. For example, Figure 5.13 (a) shows the minimum resonant frequency (RF) with its

respective efficiency (E) that was found for the search MOHEO-1000-1 shown in Figure 5.7. This figure also identifies the end segment lines that potentially can add an adjacent segment. From these points, only segments 31 and 37 are able to annex a new segment without generating a circuit. Figure 5.13 (b) shows the result when segments 32 is incorporated, starting from segment 37. Here it can be seen that the resonant frequency decreases, but also there is a decrease in efficiency. A better resonant frequency is achieved when segment 9 is added as an extension of segment 31 (see Figure 5.13 (c)). This improvement is because the incorporation of segment 9 achieves a longer continuous line than the incorporation of the segment 32.

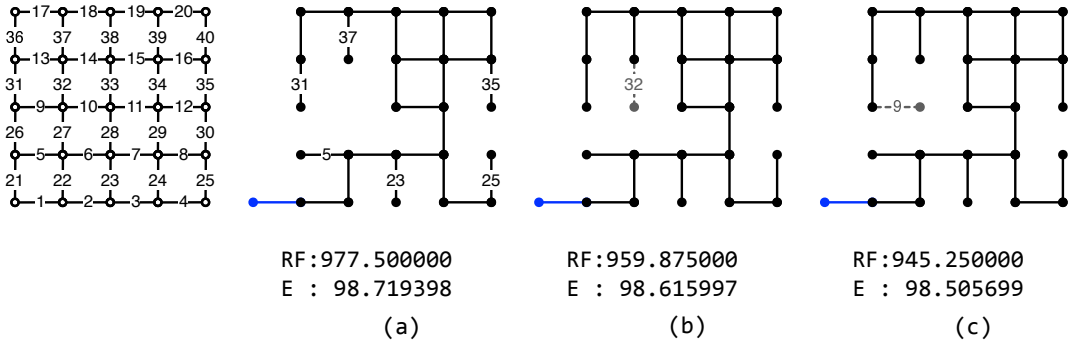


Figure 5.13: A local search example for the 5×5 grid size, when it is possible to add a new segment to an end segment line without generating a circuit. The blue segment represents the connection line.

A special case can be given here, it is not possible to add an adjacent segment without generating a circuit. Figure 5.14 (a) illustrates this specific case for the solution with the lowest resonant frequency achieved for the MOHEO approach. Despite the fact that it is not possible to add a new segment, there is the option of removing any of the branched segment lines as can be seen in Figure 5.14 (b). The elimination of one of these branched segment lines achieved a fall in the resonant frequency. This is an interesting finding that requires more analysis; however, this was not the only discovery. If, by eliminating segment 40, segments 5 and 9 are incorporated into the circuit that is generated at the end of the continuous segment starting from the line of connection, then it is possible to reduce the resonant frequency even more with a slight increase in efficiency, which is desirable (see Figure 5.14 (c)). For these two last cases, it is necessary to study a general local search heuristic that allows the discovery of new non-dominated points towards lower resonant frequencies for the antenna design generated by the MOHEO approach.

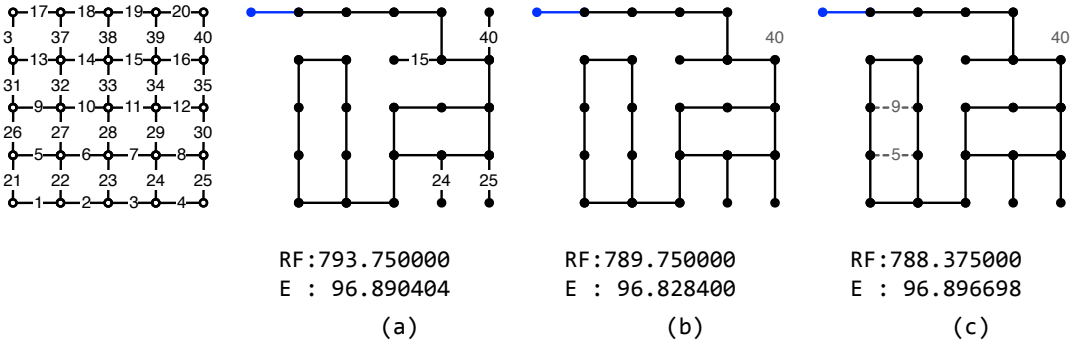


Figure 5.14: A local search example for the 5×5 grid size, where it is not possible to add a new segment to an end segment line without generating a circuit. The blue segment represents the connection line.

A third interesting case to discuss is given by analysing the solution with the lowest resonant frequency obtained by MOHEO for the grid size 6×6. Figure 5.15 (a) shows the antenna design and identifies the end segment lines that potentially could annex an adjacent segment. From these segments, the only one that can add a new segment without generating a circuit is segment 16. When segment 43 is annexed to segment 16, the resonant frequency has a slight decrease from 969.625 to 968.875 (see Figure 5.15 (b)). However, if the connection line is considered in the local search process, the simple act of changing the connection point to a new position could achieve a significant improvement in the reduction of the resonant frequency. Figure 5.15 (c) shows the considerable fall in the resonant frequency from 968.875 to 741.75 when the connection line was moved from segment 21 to segment 55.

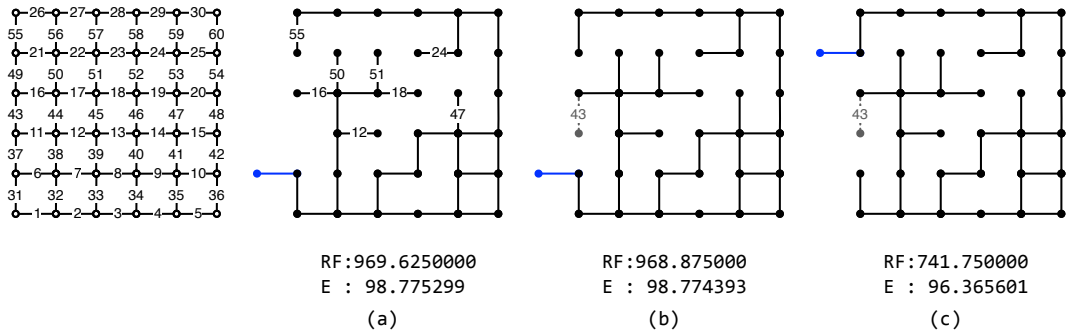


Figure 5.15: A local search example for the 6×6 grid size, given a change to the connection line to a different segment in the symmetrical line side. The blue segment represents the connection line.

As can be seen, the incorporation of a local search mechanism to improve the convergence of the antenna design generated by the MOHEO approach is a task that requires some time to be studied. This is with the objective of developing a mecha-

nism that needs the least number of assessments as possible due to the considerable amount of time required for the NEC software to perform such evaluations. Also, it must be taken into account, the secondary search mechanism present in the MOHEO framework may apply the local search operator either every time a feasible solution is generated or activate it at pre-set intervals throughout the MOHEO search process in a periodical way. Thus, the local search could multiply the number of solutions generated to be evaluated by the NEC software, by a factor determined by how many additional solutions are generated by it.

Independently of the incorporation of a local search mechanism, there are two additional aspects that deserve to be considered in the improvement of the search process. The first has to do with the definition of how the initial solution is configured. Currently, the initial solution is randomly generated and this leads to the probability that the MOHEO process has to find non-dominated solutions from an initial complex antenna design. This might take some extra time in evaluation that can be avoided if the search starts from a known solution that is considered as a good starting point. Interesting research in this regard can be found in the work developed by Weis et al. [301]. The second issue is in relation to the multi-objective pheromone-based fitness evaluation. The MOHEO approach developed until now works with a pheromone structure that rewards good solution components of the non-dominated solutions that have been found by the search process. However, the antenna designs with a high efficiency are easier to produce than those with a low resonant frequency. For this reason, the pheromone level for the components involved in an antenna design with a good efficiency will be more advantaged than the components involved in an antenna design with a low resonant frequency. For this reason, it is believed that one way to balance this inequality is to work with separated pheromone structures, one for each objective, and use an aggregating function scheme with a weighted-sum mechanism to generate the multi-objective fitness evaluation for the components of the solution. These weights could be assigned either in a static or dynamic way, with the aim of balancing the search towards low frequencies solutions.

Finally the last aspect to be considered, but not the least important, is related to the implementation of the MOHEO approach using parallel computing. Having the possibility of performing a number of evaluations simultaneously can save a lot of wall-clock time. This could be very favourable in the implementation of some local search mechanism where a number of potential alternatives could be evaluated at the same time by different processors. All this is contemplated due to the very time-consuming process that is performed by the NEC software.

5.5.2 Comparison of Results and Discussion

Results from the MOHEO were compared with those from the earlier ACO used by Lewis et al. in [193, 192] and from the recent DE used by Montgomery et al. [216]. These variants are denominated as ACO_{q1}, ACO_{q5}, ACO_{q9}, DE, DE_{minL}. The q index for the ACO approaches refers to three different levels of greediness; that is, the probability that a greedy decision is made instead of a probabilistic one. The DE alternative $minL$ alludes to the incorporation of a minimum length constraint in the meander line to encourage the exploration of solutions with lower resonant frequencies.

The objective of this comparative study is to discuss the strengths and weaknesses of the MOHEO approach and the contribution that this new technique in the RFID antenna design can provide. To do this, it is necessary to consider the implementation differences that affect this comparison, which are detailed in the following list:

- The most radical difference about the current research and the previous works is given by the design of the antenna. While the ACO and DE approaches work under the restriction of generating meander lines, which can have different lengths, the MOHEO algorithm develops an antenna based on the presence or absence of the segments that make up the grid. This gives to the MOHEO approach, the freedom to create any design. The relaxation in the meander line antenna design is based on the question of whether MOHEO is able to find new non-dominated solutions using a different antenna model that generates a modified meander line. This difference must be kept in mind at the moment of analysing the results.

- The connection line, which is the the link connecting the two halves of the dipole antenna, is another aspect that has been dealt with in different ways. The DE approach always makes the connection at node 1, while the ACO algorithm may make the connection in any node on the symmetrical line. For both cases, this connection point becomes the starting point for the meander line. As the antenna design for the MOHEO approach does not have a shape, the connection point can be assigned to any point in the symmetrical line where there is an allocated segment on the grid. If there is more than one point where the connection can be made, then it is chosen at random within the set of feasible points.

- The local search mechanism, used to complement the search process, has only been implemented by the ACO approach through the application of the backbite operator. The DE and MOHEO algorithms have not yet implemented a local search mechanism.

- The number of solutions produced by DE and ACO were 10100 and 10000 solutions respectively. However, MOHEO produced only between 1000 and 3000 solutions according to the performed experiment, due to the limited time and available computational resources. Both, DE and ACO, are population-based techniques and the size of the population for each approach was set to 100 individuals, for DE, and 10 ants, for ACO. The variability of new solutions generated by population-based methods is generally greater than the single-individual-based methods, as is the case for extremal optimisation. Hence, each new solution in MOHEO has a single change in one segment on the grid.

Taking account of these considerations, Figures 5.16, 5.17, 5.18, 5.19, 5.20 and 5.21 show the plotted graph results for the ACO, DE and MOHEO approaches for the grid sizes from 5×5 to 10×10. From these graphs, the first issue that can be appreciated is an approximated Pareto-front set that is biased towards the objective related to the efficiency. This can be interpreted as the modified meander line antennas that are created by the MOHEO approach generate designs with a high efficiency, in a natural way, instead of antenna designs with low resonant frequencies. This is evident by observing the number of solutions generated in the frequency range 1 GHz to 2 GHz. Furthermore, most of the solutions generated in that frequency range achieved better efficiency than those generated by the approaches used in previous works.

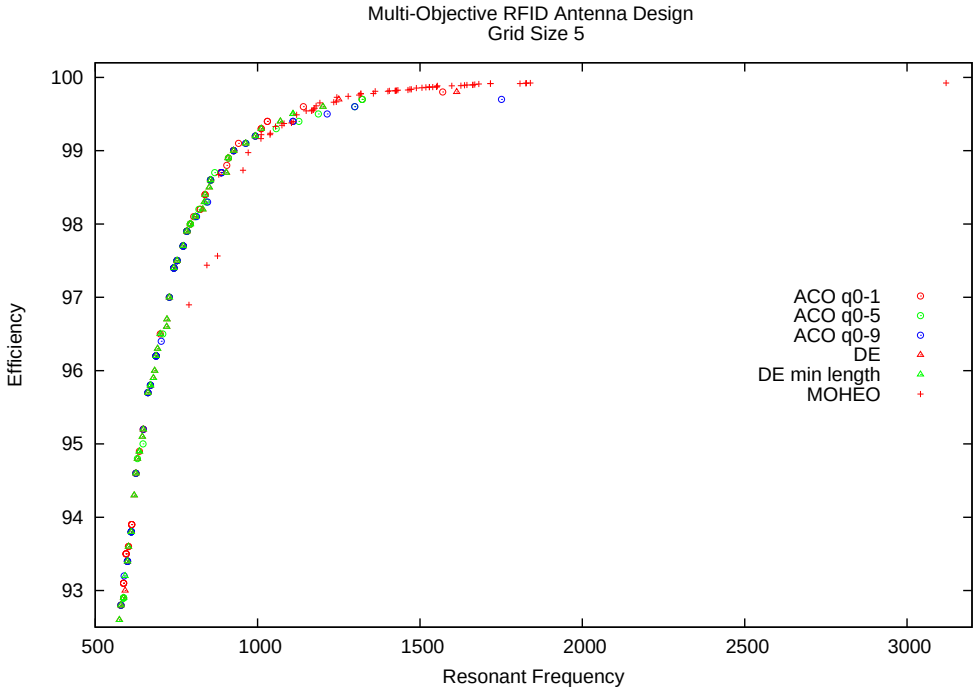


Figure 5.16: Result for the 5×5 bi-objectives RFID antenna design problems.

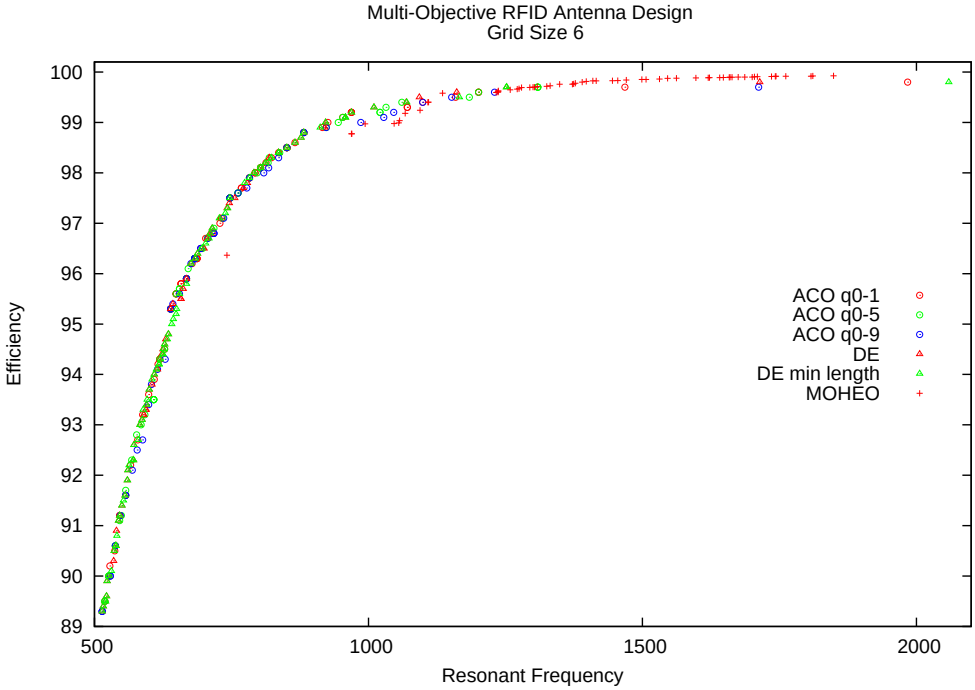


Figure 5.17: Result for the 6×6 bi-objectives RFID antenna design problems.

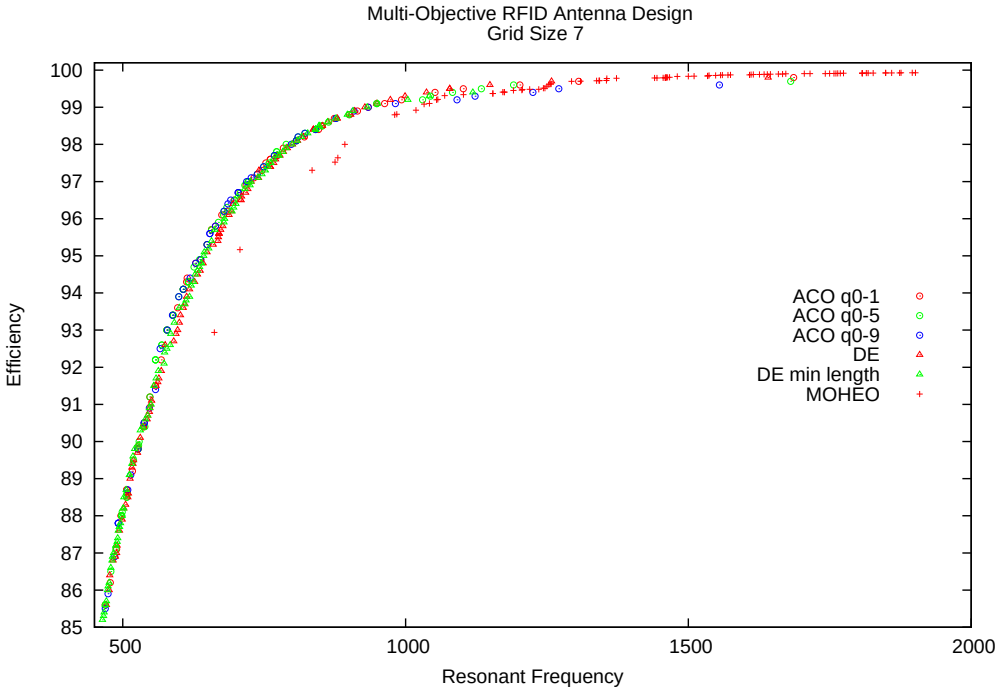


Figure 5.18: Result for the 7×7 bi-objectives RFID antenna design problems.

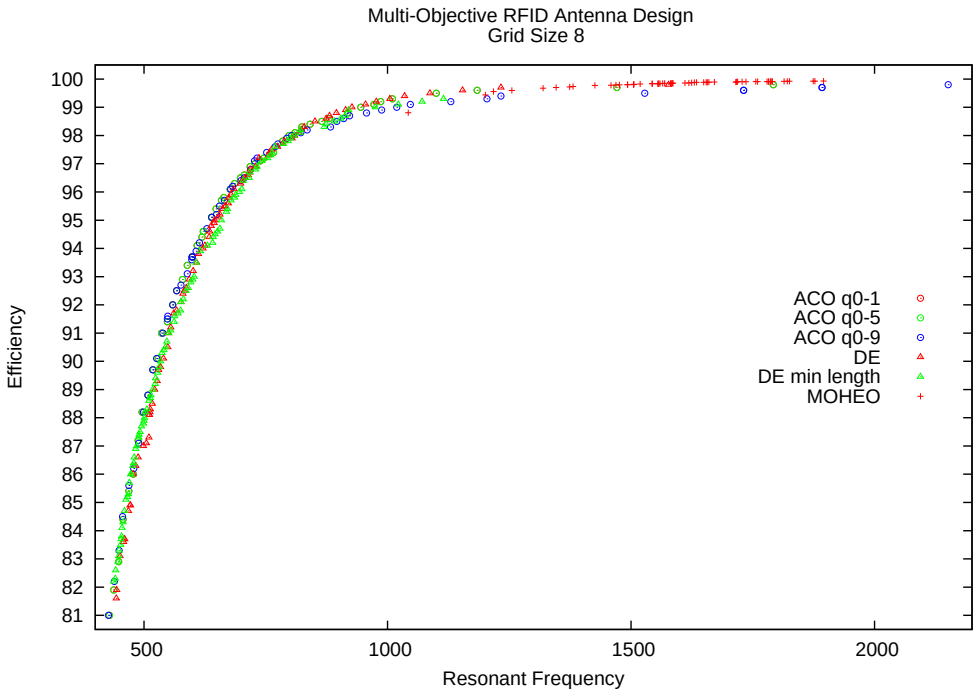


Figure 5.19: Result for the 8×8 bi-objectives RFID antenna design problems.

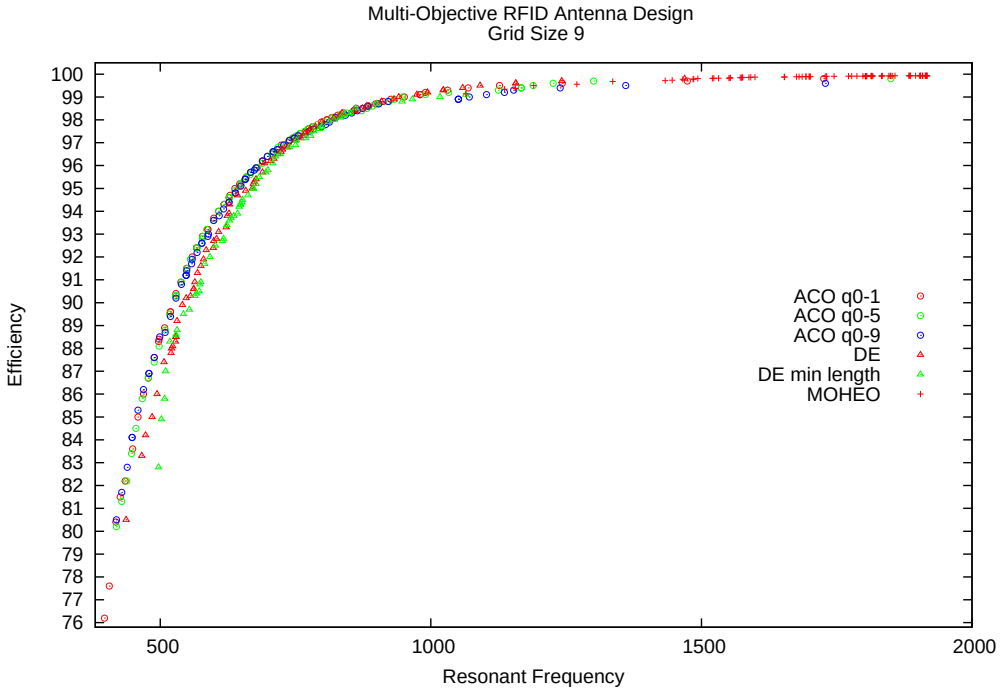


Figure 5.20: Result for the 9×9 bi-objectives RFID antenna design problems.

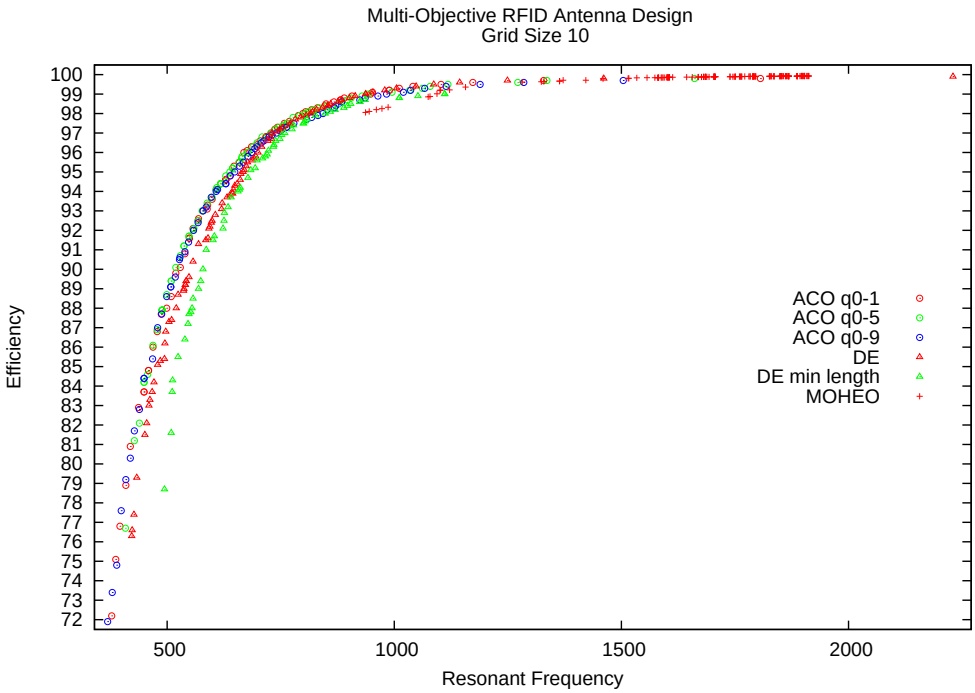


Figure 5.21: Result for the 10×10 bi-objectives RFID antenna design problems.

Use of the S-metric and the C-metric can achieve a more analytical comparison than the visual analysis of the plotted graphs. The S-metric evaluates how much of the multi-objective space is dominated by a given approximated Pareto-front set. This metric is frequently used to measure, in an independent way, the coverage of an attainment surface. A high S-metric value means a wide coverage of the approximated Pareto-front set under analysis. The C-metric compares two approximated Pareto-front sets through measuring the proportion of points in one set that are dominated by the other set. Thus, the value $C(A, B) = 1$ means that all solutions in B are dominated by A. The value $C(A, B) = 0$ represents the situation where none of the solutions in B are dominated by A. Both orderings have to be considered since $C(A, B)$ is not necessarily equal to $1 - C(B, A)$.

From Table 5.2 it can be observed that MOHEO achieved a similar coverage of the dominated space in comparison with the other approaches for the grids of size 5×5, 6×6 and 7×7. However, for the grids of size 8×8, 9×9 and 10×10, the S-metric indicated a lower dominated space than the ACO and DE approaches. The decrease in the value of the metric for the three largest grids was due to the reduced number of iterations that MOHEO carried out (just 1000, versus 3000 performed by the three smaller grids). This demonstrates that MOHEO is able to find new, non-dominated solutions as the number of iterations is increased. It is important to note that despite

the similarity of the S-metric value achieved by MOHEO, the approximated Pareto-front sets obtained by MOHEO is more inclined towards higher resonant frequencies than lower ones. For example, a visual inspection for the 6×6 grid in Figure 5.17 can illustrate this trend. This topic was discussed above, concerning how this trend may be reversed.

	S-Metric					
Grid Size	ACO_{q1}	ACO_{q1}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
5×5	49331.8	48658.8	48600.0	49326.1	48053.9	49422.6
6×6	86862.4	86557.3	86081.1	86949.5	86898.8	86319.2
7×7	48587.1	48333.5	48005.2	48583.5	47525.4	47118.1
8×8	27738.0	27738.0	27473.7	27684.6	27201.7	19713.8
9×9	48250.1	48177.5	47847.0	47835.6	46291.9	36255.8
10×10	47477.0	47333.5	47303.5	46678.4	44753.3	35810.6

Table 5.2: The S-metric values for the RFID antenna design with a grid size of 5×5, 6×6, 7×7, 8×8, 9×9 and 10×10.

The C-metric values shown in Table 5.3 illustrate that even with the low number of iterations performed by the MOHEO algorithm, the proportion of solutions found for this, that are dominated by the other approaches, is less than 32%, even becoming zero as is the case in the 9×9 grid. These values can be considered to be good, but it is necessary to develop a deeper analysis to understand why these values are received. A possible reason for these results is because of the fact that the MOHEO approach creates a type of antenna with different features to those generated by the previous works.

While both the ACO and DE algorithms were focussed on meander line antenna design, MOHEO was implemented to generate a modified meander line antenna. The former always produce a single line that can have the shape of a serpentine, meander or plough path. The main consideration to keep in mind for this type of design is to try to build the line with the longest possible length. Thus, the greater the line length, the lower the resonant frequency. The latter creates a modified meander line, which can have loops, or mesh segments (which should increase the antenna's inductive load, a desirable feature in, for example, RFID applications [120]) and parasitic elements. The inductive load stores energy that is subsequently used by the tag to transmit the radio wave signal back to the reader.

For this reason, the MOHEO approach generates a greater number of solutions with higher resonant frequencies and efficiencies than the ACO and DE algorithms, which produce a larger number of antennas with lower resonant frequencies and effi-

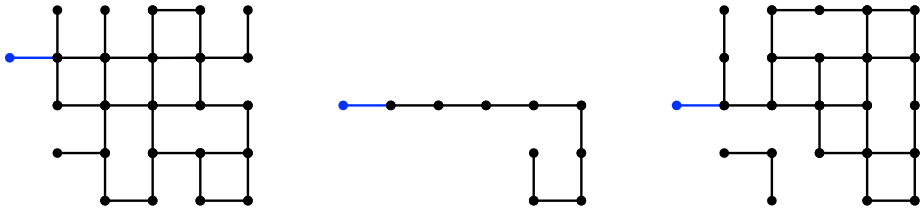
ciencies. It is believed that these two forms of production of antennas can complement each other so as to produce a more varied range of designs, which could be used for different purposes.

C-Metric

5 × 5	ACO_{q1}	ACO_{q5}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
ACO_{q1}	-	0.290909	0.347826	0.222222	0.177778	0.279412
ACO_{q5}	0.083333	-	0.152174	0.044444	0.044444	0.102941
ACO_{q9}	0.0	0.036364	-	0.044444	0.044444	0.102941
DE	0.027778	0.2	0.173913	-	0.0	0.205882
DE_{minL}	0.013889	0.163636	0.152174	0.044444	-	0.205882
$MOHEO$	0.027778	0.109091	0.065217	0.066667	0.022222	-
6 × 6	ACO_{q1}	ACO_{q5}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
ACO_{q1}	-	0.304348	0.442308	0.131579	0.219512	0.161765
ACO_{q5}	0.146667	-	0.442308	0.171053	0.243902	0.161765
ACO_{q9}	0.08	0.130435	-	0.105263	0.134146	0.147059
DE	0.333333	0.391304	0.519231	-	0.207317	0.279412
DE_{minL}	0.293333	0.391304	0.480769	0.157895	-	0.25
$MOHEO$	0.04	0.072464	0.038462	0.013158	0.02439	-
7 × 7	ACO_{q1}	ACO_{q5}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
ACO_{q1}	-	0.285714	0.229167	0.55914	0.397959	0.301075
ACO_{q5}	0.245614	-	0.208333	0.623656	0.479592	0.290323
ACO_{q9}	0.210526	0.116883	-	0.537634	0.397959	0.107527
DE	0.333333	0.181818	0.1875	-	0.173469	0.311828
DE_{minL}	0.315789	0.233766	0.25	0.731183	-	0.161290
$MOHEO$	0.017544	0.025974	0.104167	0.010753	0.0	-
8 × 8	ACO_{q1}	ACO_{q5}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
ACO_{q1}	-	0.0	0.517241	0.689655	0.656566	0.066667
ACO_{q5}	0.0	-	0.517241	0.689655	0.656566	0.066667
ACO_{q9}	0.173077	0.173077	-	0.655172	0.585859	0.016667
DE	0.269231	0.269231	0.344828	-	0.474747	0.083333
DE_{minL}	0.057692	0.057692	0.241379	0.333333	-	0.016667
$MOHEO$	0.038462	0.038462	0.137931	0.011494	0.0	-
9 × 9	ACO_{q1}	ACO_{q5}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
ACO_{q1}	-	0.573333	0.608108	0.824324	0.982759	0.051282
ACO_{q5}	0.166667	-	0.540541	0.72973	0.965517	0.038462
ACO_{q9}	0.102564	0.16	-	0.662162	0.844828	0.0
DE	0.166667	0.253333	0.297297	-	0.87931	0.102564
DE_{minL}	0.012821	0.04	0.148649	0.094595	-	0.0
$MOHEO$	0.025641	0.013333	0.094595	0.0	0.017241	-
10 × 10	ACO_{q1}	ACO_{q5}	ACO_{q9}	DE	DE_{minL}	$MOHEO$
ACO_{q1}	-	0.408451	0.701493	0.886364	1.0	0.112782
ACO_{q5}	0.39726	-	0.656716	0.806818	1.0	0.105263
ACO_{q9}	0.123288	0.098592	-	0.647727	0.95	0.090226
DE	0.109589	0.197183	0.38806	-	1.0	0.127820
DE_{minL}	0.0	0.0	0.014925	0.0	-	0.052632
$MOHEO$	0.013699	0.014085	0.029851	0.011364	0.033333	-

Table 5.3: The C-metric values for the RFID antenna design with a grid size of 5×5, 6×6, 7×7, 8×8, 9×9 and 10×10.

Next, Figures 5.22 and 5.23 show the structure of the antenna segments with a resonant frequency in the range of the desired operating frequency for the solutions with the highest efficiency in the 5×5 grid. In these two examples, $MOHEO$ achieved better solutions than ACO_{q1}. Hence, an interesting exercise is to observe the differences in the design of the antennas.



MOHEO
RF:1601.119995
E : 99.877602

(a)

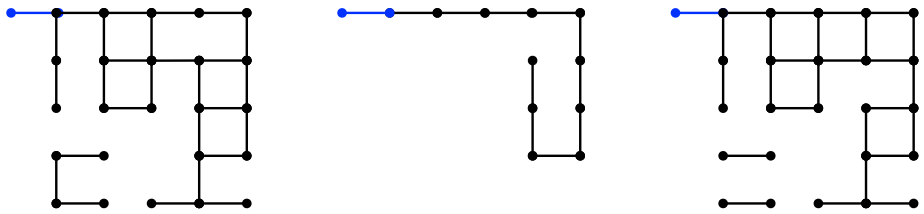
ACOq1
RF:1570.000000
E : 99.800000

(b)

MOHEO
RF:1529.119995
E : 99.867996

(c)

Figure 5.22: Structure of the 5×5 antenna segments with highest efficiency in a resonant frequency around 1.57 GHz. The blue segment represents the connection line.



MOHEO
RF:1362.119995
E : 99.810799

(a)

ACOq1
RF:1322.000000
E : 99.700000

(b)

MOHEO
RF:1318.250000
E : 99.776901

(c)

Figure 5.23: Structure of the 5×5 antenna segments with highest efficiency in a resonant frequency around 1.32 GHz. The blue segment represents the connection line.

From both figures it can be observed that the ACO antenna designs are either completely or partially included in the MOHEO antenna design. Furthermore, the number of segments, in the minimum path from the feed point for the dipole to the furthest point that can be reached, are quite similar between both approaches. However, it can be appreciated that there is a considerable difference in the total number of segments used by each algorithm. While ACO uses a reduced number of segments (just the necessary ones to build the continuous line), the MOHEO approach

allocates a higher number of segments. This additional number of segments used in the antenna design, modifies the original shape of the meander line by adding branches forming loops and meshes. The incorporation of these new elements to the meander line results in an increase on the inductive load, which benefits the efficiency of the antenna. Comparison of the antenna shapes that were generated by both methods require deeper and developed study, which will be conducted in future work with the support of experts in the design of RFID antennas.

5.6 Summary

This chapter described the implementation of an algorithm based on the multi-objective hybrid extremal optimisation framework, which was applied to solve a real-world RFID antenna design problem.

Firstly, a study of the issues involved in the operation and creation of RFID antennas was carried out. Furthermore, the previous works developed in this area were reviewed so as to have a solid foundation for MOHEO applied to this problem. At this stage, the need to extend the differentiated fitness evaluation scheme to deal with problems that have no available information or data to perform the fitness evaluation for the components of the solution was evident. In response to this requirement, the inseparable fitness evaluation technique was proposed. The implementation of this technique used a pheromone-based mechanism (borrowed from ant colony optimisation) to assess the contribution of the components of the solution.

Once the multi-objective hybrid extremal optimisation framework was suitably adjusted, the next step was to apply it to solve some benchmark grid sizes for RFID antenna design. Here, a new representation of the solution based on the allocation of segment lines was implemented. With this novel representation, the RFID antenna design was set out as a knapsack problem, which allowed the generation of a new type of antenna design through a modified meander line.

In the computational experiments stage, the MOHEO algorithm was tested with a set of six benchmark grid sizes, the same as those already used in previous works. The generated results were used to analyse the performance of the MOHEO approach, and to compare with the previously implemented proposals based on ant colony optimisation and differential evolution.

Results show that the MOHEO algorithm is an effective and efficient mechanism to find antenna designs with high efficiency for resonant frequencies over 1 GHz. The new scheme of the modified meander lines generated by the MOHEO approach have

features that the traditional meander line does not have, such as the increase in the antenna inductive load. However, it is believed that these same features prevent the new approach from achieving low resonant frequency antenna designs, which are efficiently produced by the ACO algorithms. For this reason, a set of mechanisms, to be implemented as extensions to the MOHEO approach, improving its convergence towards antenna design with low resonant frequencies, were proposed.

Despite the low number of evaluations that could be performed with the MOHEO algorithm (between 1000 and 3000 evaluations, depending on the size of the instance) compared with the 10000 evaluations performed by previous works, the preliminary results are very promising. A number of potential future works emerge from the proposal developed in this chapter. Firstly, a population-based extension of the MOHEO approach is an interesting aspect to be developed, with the aim of increasing the diversity of the solutions generated by MOHEO in the search space. This improvement will be the basis for a parallel implementation of the MOHEO algorithm, with the objective of counteracting the considerable wall-clock time required by the NEC evaluation software. Finally, taking into consideration the good performance achieved by MOHEO to find antenna designs with high efficiencies and by ACO to find antenna designs with low resonant frequencies, a new hyper-heuristic that uses both methods could be implemented. It is believed that with this new hyper-heuristic, enhanced approximated Pareto-front sets could be found.

5.7 Contributions

Based on the theoretical development and the results of the experimentation phase, it is evident that the following two objectives have been achieved:

- To present an inseparable fitness evaluation technique for the extremal optimisation heuristic to handle problems that do not provide the necessary information to perform a separable evaluation of components.
- To present an initial extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for a real-world multi-objective case.

With the fulfilment of these objectives, it is believed that the work developed in this chapter has contributed to the knowledge of fitness evaluation, extremal optimisation, hybrid methods, multi-objective optimisation for CCOPs and RFID antenna design in the following ways:

- The inseparable fitness evaluation technique is proposed as a novel fitness evaluation method for extremal optimisation. This technique allows extremal optimisation to perform fitness evaluations for those problems that do not provide the necessary information to perform a separable evaluation of components; i.e. are black box functions.

- The pheromone-based mechanism used to assess the contribution of the components of the solution in the inseparable fitness evaluation scheme. The incorporation into extremal optimisation of this mechanism used in other nature-inspired meta-heuristics is a relevant contribution to hybrid methods. Hence, the MOHEO framework is able to enhance its capability in order to solve other multi-objective combinatorial problems through the incorporation of new specialised operators.

- The application of MOHEO to a real-world problem has been demonstrated. This corroborates that it is possible to incorporate new operators to the MOHEO framework so as to successfully solve unexplored real-world problems. This offers a new approach to be used in the field of multi-objective optimisation problems.

- A new application for the RFID antennas design is provided. The implementation of the approach developed in this chapter provides a novel tool based on nature-inspired mechanisms, for the automatic optimisation of RFID antenna design through modified meander lines.

- The incorporation of new features to extremal optimisation, which can extend the spectrum of optimisation problems in which the extremal optimisation technique is able to be applied.

Chapter 6

Conclusions and Further Work

6.1 Summary

Extremal optimisation is a relatively recent nature-inspired meta-heuristic that is characterised by its simplicity of implementation and processing. That is, this technique does not require complex operators to explore the search space. Despite having these desirable characteristics, extremal optimisation has been a comparatively understudied method. Most of the applications of extremal optimisation have been in the area of unconstrained single-objective optimisation. This is because this technique was not originally designed to solve, in a generic way, a wide range of different types of problems. However, noting the present features in extreme optimisation, this thesis postulates that this meta-heuristic can be applied successfully to single-objective and multi-objective constrained combinatorial optimisation problems. To accomplish this, a series of extensions to the canonical extremal optimisation must be carried out.

In the first part of this work, the core aspects of extremal optimisation were improved. Taking into consideration that canonical extremal optimisation is only able to solve problems where the solution representation exclusively generates feasible solutions, a constraint-handling mechanism to address optimisation problems where the solution state is allowed to move between feasible and infeasible regions, was developed. This scheme was implemented through a component evaluation of the solution, which assessed the level of optimality or violation of constraints using the concept of shadow price. After that, a secondary search mechanism to improve the convergence of solutions was proposed. Here, the general double traverse local search algorithm was proposed as an initial secondary search approach. These new extensions led to the definition and implementation of an initial framework based on extremal optimisation, which used a hybrid methodology to solve constrained combinatorial problems.

Both the theoretical analysis and the empirical computational experiments, on a set of benchmark problems, showed the applicability and competitiveness of the hybrid extremal optimisation framework to solve single-objective constrained combinatorial optimisation problems.

The second stage implemented a series of modifications on the extensions that were developed in the first part, with the aim of processing multi-objective optimisation problems. Here, the differentiated fitness evaluation scheme was supplemented with an aggregating function technique to measure the level of optimality or violation of constraints for the components of the solution, when two or more objective functions must be optimised. Furthermore, the double traverse local search algorithm was complemented with a modified weighted scalar function, which was hybridised with concepts borrowed from the lexicographic ordering technique. This new multi-objective version of the double traverse local search allowed finding new non-dominated points towards the ends of the approximated Pareto-front set and also to improve the convergence of these. The computational experiments carried out on three well-known multi-objective optimisation problems produced interesting results. On one hand, when the problem had all the necessary information to perform an appropriate evaluation of the components of the solution, the MOHEO approach was able to find either competitive or similar results compared to standard techniques in the literature. On the other hand, when information provided by the problem was not sufficient or adequate to perform an appropriate assessment of the components of the solution, the MOHEO approach found it difficult to converge towards the objective function(s) that has/have the deficiency of information. Despite this drawback, the overall results obtained by MOHEO are preliminary, but promising.

In the third and final stage of this thesis, a real-world problem related to the design of RFID antennas, was addressed. This problem presented an interesting challenge due to the total lack of information about the evaluation process of the solution, which was performed by the specialised antenna evaluation software NEC (ostensibly a black box process). The main work of this part was the implementation of a mechanism that would allow the assessment of the contribution of the components of the solution in an indirect way. More specifically, a way to calculate the fitness for each component of the solution, when the problem to be solved did not provide the necessary information to carry out a separable evaluation for each component, was found. Here, the differentiated fitness evaluation scheme was complemented with an inseparable fitness evaluation technique based on the pheromone structure that is used by ant colony optimisation. The level of pheromone was updated each time a non-dominated point

was found, increasing the level of the component that was present in the solution and decreasing the level for those that were not. The pheromone level for each component of the solution was considered as the fitness evaluation for the extremal optimisation process. Results achieved by the MOHEO approach were compared with those obtained by previous works using either ACO or DE meta-heuristics. The results showed that the type of antenna design generated by MOHEO was better with respect to the efficiency, however the range of resonant frequencies reached were somewhat limited. However, this drawback in the resonant frequency objective could be improved by applying a local search mechanism, as was demonstrated by a study conducted manually in this regard in Chapter 5, for a later implementation of this.

6.2 Contributions

The work developed in this thesis has contributed to the knowledge of extremal optimisation constraint handling hybrid methods for single-objective and multi-objective CCOPs and RFID antennas design in the following ways:

- The differentiated fitness evaluation scheme is proposed as a novel constraint handling technique for extremal optimisation. This scheme allows extremal optimisation to solve not only problems where the representation of the solution generates exclusively feasible solutions, but also those that generate infeasible solutions.

- A component fitness evaluation based on the level of optimality or violation of constraints is established to complement the differentiated fitness evaluation scheme. This provides a formal criterion to evaluate each component of the solution instead of evaluating the complete solution (a mechanism widely used in evolutionary algorithms).

- The multi-objective differentiated fitness evaluation scheme is developed as an extension of the single-objective version for extremal optimisation. This extension allows extremal optimisation to solve single and multi-objective problems, with or without constraints, for the representation of solutions that generate either exclusively feasible solutions or infeasible solutions.

- The inseparable fitness evaluation technique is proposed as a novel fitness evaluation method for extremal optimisation. This technique allows extremal optimisation to perform fitness evaluations for those problems that do not provide the necessary information to perform a separable evaluation of components.

- The pheromone-based mechanism used to assess the contribution of the components of the solution in the inseparable fitness evaluation scheme. The incorporation into extremal optimisation of this mechanism used in other nature-inspired meta-heuristics is a relevant contribution to hybrid methods. Hence, the MOHEO framework is able to enhance its capability in order to solve other multi-objective combinatorial problems through the incorporation of new specialised operators.

- A generalisable secondary search mechanism based on memetic algorithms is incorporated into the extremal optimisation meta-heuristic as a method to improve the convergence of solutions. This diversifies the way that the search is performed in the neighbourhood of the last feasible solution which was found to improve the convergence of HEO.

- The double traverse local search algorithm is proposed as a general secondary search mechanism. This algorithm develops a fine-grained search in the neighbourhood of the last feasible solution found, taking advantage of the representation of the solution used in HEO. Thus, it is the first proposal of several future mechanisms that will be incorporated as the secondary search mechanism.

- The multi-objective double traverse local search algorithm is developed as an extension of the single objective version. This algorithm incorporates a modified weighted scalar fitness, which is merged with the lexicographic ordering method. As a result, a mechanism that allows the single solution in EO to travel towards the ends of the approximate Pareto-front set in a nomadic way, is achieved. That is, the single-solution permanently walks on the landscape looking for better non-dominated points.

- A hybrid extremal optimisation framework is generated as an improvement to the extremal optimisation meta-heuristic. This allows the exploration capacity of HEO to be enhanced.

- A multi-objective hybrid extremal optimisation framework is generated as an improvement to the hybrid extremal optimisation framework. This is the first multi-objective version based on the canonical extremal optimisation meta-heuristic to solve multi-objective CCOPs. Thus, MOHEO responds to the need for implementing a multi-objective version of EO as was discerned in Section 2.3.2.2.

- An exploratory and novel hybrid method based on extremal optimisation is incorporated into the nature-inspired algorithm to offer the possibility of solving capacitated constrained combinatorial optimisation problems. This permits the nature-inspired search scheme used by extremal optimisation to be applied to a new range of combinatorial optimisation problems. Also, this gives the opportunity for new CCOPs be solved by a novel meta-heuristic, like extremal optimisation, with the possibility of finding more efficient and effective results.

- A new application for the RFID antennas design is provided. The implementation of the approach developed in Chapter 5 provides a novel tool based on nature-inspired mechanisms to the automatic optimisation of the RFID antennas' design through modified meander lines.

- The incorporation of new features to extremal optimisation extends its applicability towards new optimisation problems. Here, the new way to evaluate solution components in black box optimisation using a pheromone scheme, allows extremal optimisation to solve unexplored real-world optimisation instances.

- A number of potential subsidiary projects emerge from the work developed in this thesis as mentioned in Section 3.5, Section 4.5 and Section 5.6. They offers the possibility to develop a more exhaustive investigation to improve the features incorporated in HEO and MOHEO in this thesis. Also, these potential projects give the opportunity to add new techniques that will enhance the EO mechanism so as to solve unexplored optimisation problems.

6.3 Conclusions and Future Work

Based on the theoretical development and the results of the experimentation phase, it is evident that in this thesis the following objectives have been achieved:

- To incorporate a constraint-handling mechanism that allows extremal optimisation to deal with infeasible solutions.

- To provide a hybrid extremal optimisation framework to solve single-objective constrained combinatorial optimisation problems.

- To present an initial extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for multi-objective cases.

- To present an inseparable fitness evaluation technique for the extremal optimisation heuristic to handle problems that do not provide the necessary information to perform a separable evaluation of components.
- To present an initial extremal optimisation framework which can be applied to solve constrained combinatorial optimisation problems for a real-world multi-objective case.

With the achievement of these objectives, the aim of this thesis proposing a novel and simple nature-inspired framework to solve constrained combinatorial optimisation problems, for both single-objective and multi-objective optimisation, based on the extremal optimisation heuristic, has been accomplished.

Therefore, the research question formulated in this thesis with respect to whether, extremal optimisation will be a competitive heuristic to solve single-objective and multi-objective constrained combinatorial optimisation problems, has been answered in the affirmative.

However, despite the experiments that were performed being extensive, they were not exhaustive. For this reason, a set of forthcoming research projects can be formulated.

- To solve more complex test problems for the MKP, BPP and GAP by applying new variants to the HEO framework.
- To solve the problems discussed in Section 3.3.4, which are: the frequency assignment problem, the capacitated vehicle routing problems, the single source capacitated facility location problem, the capacitated minimal spanning tree problem and the capacitated set covering problem.
- To develop a detailed study about how much of the overall results obtained by the proposed HEO framework are due to the local search mechanism. To accomplish this, it is necessary to carry out two new experiments. The first is to perform a comparison of the previous techniques used in Chapter 3 without local search to determine the level of native search power that HEO possesses in comparison to the other techniques. The second consists of analysing the performance of the same techniques when they are normalised by replacing the particular local search mechanism used for each one by a standard local search.
- To develop a population-based extension of HEO and MOHEO with the objective of improving the performance of the current frameworks. This idea is a promising avenue of future research.

- To undertake a deeper investigation about the novel random pastoralism route used by MOHEO to find the approximated Pareto-front set. This with the aim of analysing its applicability by other techniques as well as determine its contribution and benefit within the search process.

- To explore other secondary search mechanism, through either, examining in more detail the role of different local search techniques, or, applying other heuristics, such as simulated annealing, tabu search, genetic algorithms, swarm intelligence, and ant colony optimisation.

- To further investigate multi-objective differentiated fitness evaluation schemes with the aim of incorporating the evaluation based on the Pareto-dominance concept, which is popularly used by many existing multi-objective evolutionary algorithms.

- To carry out a new series of experiments to solve more complex test problems for MOKP, MOQAP and MOPFSSP by applying new variants to the MOHEO frameworks.

- To solve new benchmark problems, in multi-objective optimisation, such as the multi-objective travelling salesman problem, the multi-objective solid transportation problems and the multi-objective shortest path problem.

- To implement a parallel version of the MOHEO applied to solve RFID antenna design, in order to reduce the runtime required to obtain the approximated Pareto-front set.

- To investigate the implementation of separate pheromone representations for each objective function in the inseparable fitness evaluation scheme, in order to improve the convergence of the attainment surface for the RFID antenna design problem.

- To develop a deeper analysis of the inseparable fitness evaluation scheme to observe its functionality and extension to other problems such as the capacitated set covering problem. Also, an interesting study to be performed is to compare the performance and competitiveness of the inseparable fitness evaluation scheme against the differentiated fitness evaluation scheme.

It is evident that the work contained in this thesis gives rise to a number of potential subsidiary projects.

Bibliography

[1] Abraham, A. Evolutionary computation. *Nature 2*, August (2005), 1–13.

[2] Abramson, D., Dang, H., and Krishnamoorthy, M. A comparison of two methods for solving 0-1 integer programs using a general purpose simulated annealing algorithm. *Annals of Operations Research 63* (1996), 129–150.

[3] Abreu, B. T., Martins, E., and Sousa, F. L. Automatic test data generation for path testing using a new stochastic algorithm. In *Proceeding of 19 Simposio Brasileiro de Engenharia de Software (SBES)* (Uberlandia, October 2005), Universidade Federal de Uberlandia, pp. 247–262.

[4] Abreu, B. T., Martins, E., and Sousa, F. L. Generalized extremal optimization: A competitive algorithm for test data generation. In *XXI Simpósio Brasileiro de Engenharia de Software* (2007), vol. 1, pp. 1–10.

[5] Abreu, B. T., Martins, E., and Sousa, F. L. Generalized extremal optimization: An attractive alternative for test data generation. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation* (New York, NY, USA, 2007), H. Lipson, Ed., ACM, p. 1138.

[6] Adeli, H., and Cheng, N.-T. Augmented Lagrangian Genetic Algorithm for Structural Optimization. *Journal of Aerospace Engineering 7*, 1 (January 1994), 104–118.

[7] Aerts, J., Korst, J., and Spieksma, F. An approximation algorithm for a generalized assignment problem with small resource requirements. Open access publications from katholieke universiteit leuven, Katholieke Universiteit Leuven, 2003.

[8] Ahmed, E., and Elettreby, M. F. On multiobjective evolution model. *International Journal of Modern Physics C 15*, 9 (October 2004), 1189–1195.

[9] Ahmed, E., and Elettreby, M. F. On combinatorial optimization motivated by biology. *Applied Mathematics and Computation 172*, 1 (January 2006), 40–48.

[10] Ahuja, R. K., Orlin, J. B., Pallottino, S., Scaparra, M. P., and Scutellà, M. G. A multi-exchange heuristic for the single-source capacitated facility location problem. *Manage. Sci. 50* (June 2004), 749–760.

[11] Alvim, A. C., Glover, F. S., Ribeiro, C. C., and Aloise, D. J. Local search for the bin packing problem. In *in Extended Abstracts of the III Metaheuristics International Conference (MIC99)* (Angra dos Reis, Brazil, July 1999), P. Hansen and C. Ribeiro, Eds., Catholic University of Rio de Janeiro, pp. 7–12.

[12] Amberg, A., Domschke, W., and VoB, S. Capacitated minimum spanning trees: Algorithms using intelligent search. *Combinatorial Optimization: Theory and Practice 1* (1996), 1–39.

[13] Back, T., Fogel, D. B., and Michalewicz, Z., Eds. *Handbook of Evolutionary Computation*. IOP Publishing Ltd., Bristol, UK, UK, 1997.

[14] Bak, P. *How nature works*. Springer-Verlag New York Inc., 1996.

[15] Bak, P., and Sneppen, K. Punctuated equilibrium and criticality in a simple model of evolution. *Physical Review Letters 71*, 24 (December 1993), 4083–4086.

[16] Bak, P., Tang, C., and Wiesenfeld, K. Self-organized criticality: An explanation of the 1/f noise. *Physical Review Letters 59*, 4 (July 1987), 381–384.

[17] Baker, B. M., and Sheasby, J. Extensions to the generalised assignment heuristic for vehicle routing. *European Journal of Operational Research 119*, 1 (1999), 147 – 157.

[18] Baker, J. E. Adaptive selection methods for genetic algorithms. In *Proceedings of the 1st International Conference on Genetic Algorithms* (Hillsdale, NJ, USA, 1985), J. J. Grefenstette, Ed., L. Erlbaum Associates Inc., pp. 101–111.

[19] Balachandran, V. *An integer generalized transportation model for optimal job assignment in computer networks*. Discussion paper. Center for Mathematical Studies in Economics and Management Science, Northwestern University, 1978.

[20] Bansal, N., Krishnaswamy, R., and Saha, B. On capacitated set cover problems. In *Approximation, Randomization, and Combinatorial Optimization* (Princeton, NJ, USA, August 2011), L. A. Goldberg, K. Jansen, R. Ravi, and J. D. P. Rolim, Eds., vol. 6845 of *Lecture Notes in Computer Science*, Springer, pp. 38–49.

[21] Bansal, N., and Pruhs, K. The geometry of scheduling. *CoRR abs/1008.4889* (2010).

[22] Bar-Noy, A., Bar-Yehuda, R., Freund, A., (Seffi) Naor, J., and Schieber, B. A unified approach to approximating resource allocation and scheduling. *J. ACM 48* (September 2001), 1069–1090.

[23] Bean, J. C., and Hadj-Alouane, A. B. A Dual Genetic Algorithm for Bounded Integer Programs. Tech. Rep. TR 92-53, Department of Industrial and Operations Engineering, The University of Michigan, 1992.

[24] Beasley, J. E. OR-Library: Distributing test problems by electronic mail. *Journal of the Operational Research Society 41*, 11 (1990), 1069–1072.

[25] Belur, S. V. CORE: Constrained Optimization by Random Evolution. In *Late Breaking Papers at the Genetic Programming 1997 Conference* (Stanford University, California, July 1997), J. R. Koza, Ed., Stanford bookstore, pp. 280–286.

[26] Berman, P., Karpinski, M., and Lingas, A. Exact and approximation algorithms for geometric and capacitated set cover problems with applications. *CoRR abs/0904.2310* (2009).

[27] Bilchev, G., and Parmee, I. Constraint Handling for the Fault Coverage Code Generation Problem: An Inductive Evolutionary Approach. In *Proceedings of the Fourth Conference on Parallel Problem Solving from Nature (PPSN IV)* (Heidelberg, Germany, September 1996), H.-M. Voigt, W. Ebeling, I. Rechenberg, and H.-P. Schwefel, Eds., Springer-Verlag, pp. 880–889.

[28] Bilchev, G., and Parmee, I. Constraint handling for the fault coverage code generation problem: An inductive evolutionary approach. In *Parallel Problem Solving from Nature ? PPSN IV*, H.-M. Voigt, W. Ebeling, I. Rechenberger, and H.-P. Schwefel, Eds. Springer. Lecture Notes in Computer Science, Vol. 1141, Berlin, Germany, 1996, pp. 880–889.

[29] Bilchev, G., and Parmee, I. C. The Ant Colony Metaphor for Searching Continuous Design Spaces. In *Evolutionary Computing. AISB Workshop. Selected Papers* (Sheffield, U.K., April 1995), T. C. Fogarty, Ed., Springer-Verlag, pp. 25–39. Lecture Notes in Computer Science No. 993.

[30] Bilchev, G., and Parmee, I. C. Constrained and Multi-Modal Optimisation with an Ant Colony Search Model. In *Proceedings of 2nd International Conference on Adaptive Computing in Engineering Design and Control*, I. C. Parmee and M. J. Denham, Eds. University of Plymouth, Plymouth, UK, March 1996.

[31] Blum, C. Ant colony optimization: Introduction and recent trends. *Physics of Life Reviews 2*, 4 (December 2005), 353–373.

[32] Blum, C., and Roli, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys. 35*, 3 (September 2003), 268–308.

[33] Boettcher, S. Extremal optimization of graph partitioning at the percolation threshold. *Journal of Physics A: Mathematical and General 32* (1999), 5201–5211.

[34] Boettcher, S. Extremal optimization: Heuristics via co-evolutionary avalanches. *Computing in Science and Engineering 2* (December 2000), 75–82.

[35] Boettcher, S. Numerical results for ground states of mean-field spin glasses at low connectivities. *Phys. Rev. B 67*, 6 (Feb 2003), 060403.

[36] Boettcher, S. Extremal optimization for Sherrington-Kirkpatrick spin glasses. *The European Physical Journal B 46* (2005), 501–505.

[37] Boettcher, S., and Frank, M. Optimizing at the ergodic edge. *Physica A: Statistical Mechanics and its Applications 367* (2006), 220–230.

[38] Boettcher, S., and Grigni, M. Jamming model for the extremal optimization heuristic. *Journal of Physics A: Mathematical and General 35* (2002), 1109–1123.

[39] Boettcher, S., and Percus, A. G. Extremal optimization: Methods derived from co-evolution. In *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference* (April 1999), pp. 825–832.

[40] Boettcher, S., and Percus, A. G. Combining local search with co-evolution in a remarkably simple way. In *Proceedings of the 2000 Congress on Evolutionary Computation* (Piscataway, NJ, 2000), IEEE Service Center, pp. 1578–1584.

[41] Boettcher, S., and Percus, A. G. Nature's way of optimizing. *Artificial Intelligence 119*, 1 (May 2000), 275–286.

[42] Boettcher, S., and Percus, A. G. Extremal optimization for graph partitioning. *Physical Review E 64* (2001), 026114.

[43] Boettcher, S., and Percus, A. G. Optimization with extremal dynamics. *Physical Review Letters 86* (2001), 5211–5214.

[44] Boettcher, S., and Percus, A. G. Extremal optimization: An evolutionary local-search algorithms. In *In proceedings of the 8th INFORMS Computer Society Conference* (September 2003).

[45] Boettcher, S., and Percus, A. G. Extremal optimization at the phase transition of the 3-coloring problem. *Physical Review E 69* (2004), 066703.

[46] Boettcher, S., and Sibani, P. Comparing extremal and thermal explorations of energy landscapes. *The European Physical Journal B - Condensed Matter and Complex Systems 44* (2005), 317–326.

[47] Bullnheimer, B., Hartl, R. F., and Strauss, C. A new rank-based version of the ant system: a computational study. Tech. Rep. POM-03/97, Institute of Management Science, University of Vienna, 1997.

[48] Burkard, R., Dell'Amico, M., and Martello, S. *Assignment Problems.* Society for Industrial and Applied Mathematics, Philadephia, PA, 2009.

[49] Burke, G., Poggio, A., Logan, J., and Rockway, J. Nec - numerical electromagnetics code for antennas and scattering. In *Antennas and Propagation Society International Symposium, 1979* (jun 1979), vol. 17, pp. 147 – 150.

[50] Captivo, M. E., Clìmaco, J. a., Figueira, J., Martins, E., and Santos, J. L. Solving bicriteria 0-1 knapsack problems using a labeling algorithm. *Comput. Oper. Res. 30* (October 2003), 1865–1886.

[51] Cattrysse, D. G., and Wassenhove, L. N. V. A survey of algorithms for the generalized assignment problem. *European Journal of Operational Research 60*, 3 (1992), 260 – 272.

[52] Cela, E. Assignment problems. *Handbook of Applied Optimization, Part II - Applications* (2002), 661–678.

[53] Chakrabarty, D., Grant, E., and Könemann, J. On column-restricted and priority covering integer programs. *Computing Research Repository abs/1003.1507* (2010).

[54] Chen, M.-R., and Lu, Y.-Z. A novel elitist multiobjective optimization algorithm: Multiobjective extremal optimization. *European Journal of Operational Research 127*, 3 (August 2008), 637–651.

[55] Chen, M.-R., Lu, Y.-Z., and Luo, Q. A novel hybrid algorithm with marriage of particle swarm optimization and extremal optimization. *Optimization Online* (June 2007).

[56] Chen, M.-R., Lu, Y.-Z., and Yang, G. Population-based extremal optimization with adaptive lévy mutation for constrained optimization. In *CIS '06: International Conference on Computational Intelligence and Security* (2006), Y. Wang, Y.-M. Cheung, and H. Liu, Eds., vol. 4456 of *Lecture Notes in Computer Science*, Springer, pp. 144–155.

[57] Chen, M.-R., Lu, Y.-Z., and Yang, G. Multiobjective optimization using population-based extremal optimization. *Neural Computing & Applications* (April 2007).

[58] Chen, M.-R., zai Lu, Y., and ke Yang, G. Multiobjective extremal optimization with applications to engineering design. *Journal of Zhejiang University - Science A 8*, 12 (November 2007), 1905–1911.

[59] Chen, Y.-W., Lu, Y.-Z., and Yang, G.-K. Hybrid evolutionary algorithm with marriage of genetic algorithm and extremal optimization for production scheduling. *The International Journal of Advanced Manufacturing Technology* (January 2007).

[60] Chu, P., and Beasley, J. Constraint handling in genetic algorithms: The set partitioning problem. *Journal of Heuristics 11*, 4 (December 1998), 323–357.

[61] Chu, P., and Beasley, J. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics 4*, 1 (June 1998), 63–86.

[62] Chu, P. C., and Beasley, J. E. A genetic algorithm for the generalised assignment problem. *Computers and Operations Research 24*, 1 (January 1997), 17–23.

[63] Chung, C.-J., and Reynolds, R. G. A Testbed for Solving Optimization Problems Using Cultural Algorithms. In *Evolutionary Programming V: Proceedings of the Fifth Annual Conference on Evolutionary Programming* (Cambridge, Massachusetts, March 1996), L. J. Fogel, P. J. Angeline, and T. Bäck, Eds., MIT Press, pp. 225–236.

[64] Coello-Coello, C. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering 191*, 11-12 (January 2002), 1245–1287.

[65] Coello-Coello, C. 20 years of evolutionary multi-objective optimization: What has been done and what remains to be done. In *Computational Intelligence: Principles and Practice*, G. Yen and D. Fogel, Eds. IEEE Computational Intelligence Society, 2006, ch. 4, pp. 73–88.

[66] Coello-Coello, C. Constraint-handling techniques used with evolutionary algorithms. In *GECCO '07: Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation* (New York, NY, USA, 2007), ACM, pp. 3057–3077.

[67] Coello-Coello, C. List of references on constraint-handling techniques used with evolutionary algorithms. In web page accessed on December 2011 and available on http://www.cs.cinvestav.mx/ constraint/, December 2011.

[68] Coello-Coello, C., and Lamont, G. B. *Applications Of Multi-Objective Evolutionary Algorithms*. Advances in Natural Computation. World Scientific Publishing Company, 2004.

[69] Coello-Coello, C., Lamont, G. B., and Veldhuizen, D. A. V. *Evolutionary Algorithms for Solving Multi-Objective Problems*, Second ed. Genetics and Evolutionary Computation. Springer, 2007.

[70] Coello Coello, C. A., and Toscano Pulido, G. A Micro-Genetic Algorithm for Multiobjective Optimization. In *First International Conference on Evolutionary Multi-Criterion Optimization*, E. Zitzler, K. Deb, L. Thiele,

C. A. C. Coello, and D. Corne, Eds. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001, pp. 126–140.

[71] Coello Coello, C. A., and Toscano Pulido, G. Multiobjective Optimization using a Micro-Genetic Algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)* (San Francisco, California, 2001), L. Spector, E. D. Goodman, A. Wu, W. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds., Morgan Kaufmann Publishers, pp. 274–282.

[72] Coffman, E. G., Garey, M. R., and Johnson, D. S. Approximation algorithms for bin packing: a survey. In *Approximation algorithms for NP-hard problems*, D. Hochbaum, Ed. PWS Publishing Co., Boston, MA, USA, 1997, ch. 2, pp. 46–93.

[73] Coffman, Jr., E. G., Garey, M. R., and Johnson, D. S. An application of bin-packing to multiprocessor scheduling. *SIAM Journal on Computing 7*, 1 (February 1978), 1–17.

[74] Coit, D. W., and Smith, A. E. Penalty guided genetic search for reliability design optimization. *Computers and Industrial Engineering 30*, 4 (September 1996), 895–904.

[75] Coit, D. W., Smith, A. E., and Tate, D. M. Adaptive Penalty Methods for Genetic Optimization of Constrained Combinatorial Problems. *INFORMS Journal on Computing 8*, 2 (Spring 1996), 173–182.

[76] Connolly, D. General purpose simulated annealing. *J Oper Res Soc 43*, 5 (05 1992), 495–505.

[77] Cormen, T. H., Leiserson, C. E., Rivest, R. L., and Stein, C. *Introduction to Algorithms, Third Edition*, 3rd ed. The MIT Press, 2009.

[78] Corne, D. W., Jerram, N. R., Knowles, J. D., and Oates, M. J. PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001)* (San Francisco, California, 2001), L. Spector, E. D. Goodman, A. Wu, W. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds., Morgan Kaufmann Publishers, pp. 283–290.

[79] Corne, D. W., Knowles, J. D., and Oates, M. J. The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization. In *Proceedings of the Parallel Problem Solving from Nature VI Conference* (Paris, France, 2000), M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, Eds., Springer. Lecture Notes in Computer Science No. 1917, pp. 839–848.

[80] Csirik, J., and Woeginger, G. J. On-line packing and covering problems. In *Developments from a June 1996 seminar on Online algorithms: the state of the art* (London, UK, 1998), Springer-Verlag, pp. 147–177.

[81] Dall, J., and Sibani, P. Faster monte carlo simulations at low temperatures. the waiting time method. *Computer Physics Communications 141*, 2 (2001), 260 – 267.

[82] Danon, L., Díaz-Guilera, A., Duch, J., and Arenas, A. Comparing community structure identification. *Journal of Statistical Mechanics: Theory and Experiment 2005*, 09 (2005), P09008.

[83] Dantzig, G. B. *Linear Programming and Extensions.* Princeton Landmarks in Mathematics and Physics. Princeton University Press, 1963.

[84] Dantzig, G. B., and Ramser, J. H. The truck dispatching problem. *Management Science 6*, 1 (1959), 80–91.

[85] Darwin, C. *On the Origin of Species by Means of Natural Selection.* Murray, London, 1859. or the Preservation of Favored Races in the Struggle for Life.

[86] DasGupta, D. *Artficial Immune Systems and their Applications.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1998.

[87] Dasgupta, D., and Michalewicz, Z., Eds. *Evolutionary Algorithms in Engineering Applications.* Springer-Verlag, Berlin, 1997.

[88] Davis, L. *Genetic Algorithms and Simulated Annealing.* Pitman, London, 1987.

[89] Davis, L. D. *Handbook Of Genetic Algorithms.* Van Nostrand Reinhold Company, New York, January 1991.

[90] Dawkins, R. *The Selfish Gene*, vol. 3. Oxford University Press, 1976.

[91] Day, R. O., Kleeman, M. P., and Lamont, G. B. Multi-Objective fast messy Genetic Algorithm Solving Deception Problems. In *2004 Congress on Evolutionary Computation (CEC'2004)* (Portland, Oregon, USA, June 2004), vol. 2, IEEE Service Center, pp. 1502–1509.

[92] Day, R. O., and Lamont, G. B. An Effective Explicit Building Block MOEA, the MOMGA-IIa. In *2005 IEEE Congress on Evolutionary Computation (CEC'2005)* (Edinburgh, Scotland, September 2005), vol. 1, IEEE Service Center, pp. 17–24.

[93] Day, R. O., and Lamont, G. B. Multiobjective Quadratic Assignment Problem Solved by an Explicit Building Block Search Algorithm - MOMGA-IIa. In *Evolutionary Computation in Combinatorial Optimization. 5th European Conference, EvoCOP 2005* (Lausanne, Switzerland, March/April 2005), G. R. Raidl and J. Gottlieb, Eds., Springer, Lecture Notes in Computer Science Vol. 3448, pp. 91–100.

[94] de Castro, L. N. Fundamentals of Natural Computing: An Overview. *Physics of Life Reviews 4* (2007), 1–36.

[95] Dean, B. C., and Swar, N. The generalized stable allocation problem. In *Proceedings of the 3rd International Workshop on Algorithms and Computation* (Berlin, Heidelberg, 2009), WALCOM '09, Springer-Verlag, pp. 238–249.

[96] Deb, K. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering 186*, 2-4 (June 2000), 311–338.

[97] Deb, K. *Multi-Objective Optimization using Evolutionary Algorithms*, First ed. Wiley-Interscience Series in Systems and Optimization. Wiley, July 2001.

[98] Deb, K. Evolutionary multi-objective optimization without additional parameters. In *Parameter Setting in Evolutionary Algorithms*, F. Lobo, C. Lima, and Z. Michalewicz, Eds., vol. 54 of *Studies in Computational Intelligence*. Springer Berlin / Heidelberg, 2007, pp. 241–257.

[99] Deb, K., Agrawal, S., Pratab, A., and Meyarivan, T. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[100] Deb, K., Agrawal, S., Pratab, A., and Meyarivan, T. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *Proceedings of the Parallel Problem Solving from Nature VI Conference* (Paris, France, 2000), M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H.-P. Schwefel, Eds., Springer. Lecture Notes in Computer Science No. 1917, pp. 849–858.

[101] Dorigo, M., and Blum, C. Ant colony optimization theory: A survey. *Theoretical Computer Science 344*, 2-3 (11 2005), 243–278.

[102] Dorigo, M., Colorni, A., and Maniezzo, V. Positive feedback as a search strategy. Tech. Rep. 91-016, Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy, 1991.

[103] Dorigo, M., and Di Caro, G. The ant colony optimization meta-heuristic. In *New Ideas in Optimization*, D. Corne, M. Dorigo, and F. Glover, Eds., Advanced Topics In Computer Science. McGraw-Hill, London, 1999, ch. 2, pp. 11–32.

[104] Dorigo, M., and Gambardella, L. M. Ant colony system: a cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on 1*, 1 (1997), 53–66.

[105] Dorigo, M., and Stützle, T. *Ant Colony Optimization*. Bradford Books. The MIT Press, July 2004.

[106] Dorronsoro Díaz, B. The vrp web. In web page accessed on December 2011 and available on http://neo.lcc.uma.es/radi-aeb/WebVRP/, March 2007.

[107] Duch, J., and Arenas, A. Community detection in complex networks using extremal optimization. *Physical Review E 72* (2005), 027104.

[108] Dudek, R. A., Panwalkar, S. S., and Smith, M. L. The lessons of flowshop scheduling research. *Oper. Res. 40* (January 1992), 7–13.

[109] Dyckhoff, H., and Finke, U. *Cutting and Packing in Production and Distribution: A Typology and Bibliography*, 1 ed. Contributions to Management Science. Physica-Verlag, Heidelberg, German, August 1992.

[110] Dyson-Hudson, R., and Dyson-Hudson, N. Nomadic pastoralism. *Annual review of anthropology 9*, 1 (1980), 15–61.

[111] Erickson, M., Mayer, A., and Horn, J. Multi-objective optimal design of groundwater remediation systems: application of the niched Pareto genetic algorithm (NPGA). *Advances in Water Resources 25*, 1 (January 2002), 51–65.

[112] Falkenauer, E. A hybrid grouping genetic algorithm for bin packing. *Journal of Heuristics 2*, 1 (June 1996), 5–30.

[113] Feltl, H., and Raidl, G. R. An improved hybrid genetic algorithm for the generalized assignment problem. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing* (New York, NY, USA, 2004), ACM, pp. 990–995.

[114] Finkenzeller, K. *RFID Handbook: Fundamentals and Applications in Contactless Smart Cards, Radio Frequency Identification and Near-Field Communication, Third Edition*, john wiley & sons, inc. ed. John Wiley and Sons, 2010.

[115] Fogel, D. B. *Evolutionary Computation: The Fossil Record*, 1st ed. Wiley-IEEE Press, 1998.

[116] Fogel, D. B. *Evolutionary computation: toward a new philosophy of machine intelligence*, third ed. IEEE Press Series on Computational Intelligence. Wiley-IEEE Press, USA, December 2005.

[117] Fonseca, C. M., and Fleming, P. J. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms* (San Mateo, California, 1993), S. Forrest, Ed., University of Illinois at Urbana-Champaign, Morgan Kauffman Publishers, pp. 416–423.

[118] Fonseca, C. M., and Fleming, P. J. An overview of evolutionary algorithms in multiobjective optimization. *Evol. Comput. 3* (March 1995), 1–16.

[119] Fourman, M. P. Compaction of Symbolic Layout using Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms* (1985), Lawrence Erlbaum, pp. 141–153.

[120] Galehdar, A., Thiel, D., and O'Keefe, S. Design methods for 3d rfid antennas located on a conducting ground plane. *Antennas and Propagation, IEEE Transactions on 57*, 2 (feb. 2009), 339 –346.

[121] Galehdar, A., Thiel, D., O'Keefe, S., and Kingsley, S. Efficiency variations in electrically small, meander line rfid antennas. In *Antennas and Propagation Society International Symposium, 2007 IEEE* (june 2007), pp. 2273 –2276.

[122] Galski, R. L. *Development of Improved, Hybrid, Parallel and Multiobjective Version of the Generalized Extremal Optimization Method and its Application to the Design of Spatial Systems*. PhD thesis, Instituto Nacional de Pesquisas Espaciais, 2006.

[123] Galski, R. L., Sousa, F. L., Ramos, F. M., and Muraoka, I. Application of a new hybrid evolutionary strategy to spacecraft thermal design. In *GECCO 2004 Workshop Proceedings* (Seattle, Washington, USA, June 2004).

[124] Galski, R. L., Sousa, F. L. D., Ramos, F. M., and Muraoka, I. Spacecraft thermal design with the generalized extremal optimization algorithm. *Inverse Problems in Science and Engineering 15*, 1 (January 2007), 61–75.

[125] Garey, M. R., and Johnson, D. S. *Computers and Intractability : A Guide to the Theory of NP-Completeness*. Series of Books in the Mathematical Sciences. W. H. Freeman & Co., New York, NY, USA, January 1979.

[126] Geiger, M. J. On operators and search space topology in multi-objective flow shop scheduling. *European Journal of Operational Research 181*, 1 (2007), 195 – 206.

[127] Goldberg, D., Deb, K., and Korb, B. Messy genetic algorithms: motivation, analysis, and first results. *Complex Systems*, 3 (1989), 493–530.

[128] Goldberg, D. E. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1989.

[129] Goldberg, D. E. Sizing populations for serial and parallel genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms* (San Francisco, CA, USA, 1989), Morgan Kaufmann Publishers Inc., pp. 70–79.

[130] Gómez-Meneses, P., and Randall, M. Extremal optimisation with a penalty approach for the multidimensional knapsack problem. In *SEAL* (2008), X. Li, M. Kirley, M. Zhang, D. G. Green, V. Ciesielski, H. A. Abbass,

Z. Michalewicz, T. Hendtlass, K. Deb, K. C. Tan, J. Branke, and Y. Shi, Eds., vol. 5361 of *Lecture Notes in Computer Science*, Springer, pp. 229–238.

[131] Gómez-Meneses, P., and Randall, M. A hybrid extremal optimisation approach for the bin packing problem. In *ACAL* (2009), K. B. Korb, M. Randall, and T. Hendtlass, Eds., vol. 5865 of *Lecture Notes in Computer Science*, Springer, pp. 242–251.

[132] Gómez-Meneses, P., Randall, M., and Lewis, A. A hybrid multi-objective extremal optimisation approach for multi-objective combinatorial optimisation problems. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010* (Barcelona, Spain, July 2010), pp. 292–299.

[133] Gómez-Meneses, P., Randall, M., and Lewis, A. A multi-objective hybrid extremal optimisation approach applied to rfid antenna design. In *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation II*, O. Schütze, C. A. Coello Coello, A.-A. Tantar, E. Tantar, P. Bouvry, P. Del Moral, and P. Legrand, Eds., vol. 175 of *Advances in Intelligent and Soft Computing*. Springer Berlin Heidelberg, 2012, pp. 431–446.

[134] Gottlieb, J. On the effectivity of evolutionary algorithms for the multidimensional knapsack problem. In *Selected Papers from the 4th European Conference on Artificial Evolution* (London, UK, December 2000), Lecture Notes in Computer Science, Springer-Verlag, pp. 23–37.

[135] Hadj-Alouane, A. B., and Bean, J. C. A Genetic Algorithm for the Multiple-Choice Integer Program. *Operations Research 45* (1997), 92–101.

[136] Hajela, P., and Lee, J. Constrained Genetic Search via Schema Adaptation. An Immune Network Solution. In *Proceedings of the First World Congress of Stuctural and Multidisciplinary Optimization* (Goslar, Germany, 1995), N. Olhoff and G. I. N. Rozvany, Eds., Pergamon, pp. 915–920.

[137] Hajela, P., and Lee, J. Constrained Genetic Search via Schema Adaptation. An Immune Network Solution. *Structural Optimization 12* (1996), 11–15.

[138] Hajela, P., and Lin, C. Y. Genetic search strategies in multicriterion optimal design. *Structural Optimization 4* (1992), 99–107.

[139] Hansen, M. P. *Metaheuristics for multiple objective combinatorial optimization*. PhD thesis, Institute of Mathematical Modelling, Technical University of Denmark, March 1998.

[140] Hansen, M. P., and Jaszkiewicz, A. Evaluating the quality of approximations to the non-dominated set. Tech. Rep. IMM-REP-1998-7, Technical University of Denmark, March 1998.

[141] Heilmann, F., Hoffmann, K. H., and Salamon, P. Best possible probability distribution over extremal optimization ranks. *EPL (Europhysics Letters) 66*, 3 (2004), 305.

[142] Hendtlass, T., Moser, I., and Randall, M. Dynamic problems and nature inspired meta-heuristics. In *E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing* (Washington DC, USA, 2006), IEEE Computer Society, pp. 111–116.

[143] Hendtlass, T., and Randall, M. Extremal optimisation and bin packing. In *SEAL* (2008), vol. 5361 of *Lecture Notes in Computer Science*, Springer, pp. 220–228.

[144] Hernández-Aguirre, A., Botello Rionda, S., Coello Coello, C. A., Lizárraga Lizárraga, G., and Mezura Montes, E. Handling Constraints using Multiobjective Optimization Concepts. *International Journal for Numerical Methods in Engineering 59*, 15 (April 2004), 1989–2017.

[145] Hoare, C. A. R. Quicksort. *The Computer Journal 5*, 1 (1962), 10–16.

[146] Hoffmann, K. H., Heilmann, F., and Salamon, P. Fitness threshold accepting over extremal optimization ranks. *Phys. Rev. E 70*, 4 (Oct 2004), 046704.

[147] Hoffmeister, F., and Sprave, J. Problem-independent handling of constraints by use of metric penalty functions. In *Proceedings of the Fifth Annual Conference on Evolutionary Programming (EP'96)* (San Diego, California, February 1996), L. J. Fogel, P. J. Angeline, and T. Bäck, Eds., The MIT Press, pp. 289–294.

[148] Holland, J. H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence.* University of Michigan Press, 1975.

[149] Homaifar, A., Lai, S. H. Y., and Qi, X. Constrained Optimization via Genetic Algorithms. *Simulation 62*, 4 (1994), 242–254.

[150] Horn, J., and Nafpliotis, N. Multiobjective Optimization using the Niched Pareto Genetic Algorithm. Tech. Rep. IlliGAl Report 93005, University of Illinois at Urbana-Champaign, Urbana, Illinois, USA, 1993.

[151] Horn, J., Nafpliotis, N., and Goldberg, D. E. A Niched Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence* (Piscataway, New Jersey, June 1994), vol. 1, IEEE Service Center, pp. 82–87.

[152] Ishibuchi, H. Benchmark multiobjective permutation flowshop scheduling problem. In web page accessed on December 2011 and available on http://www.ie.osakafu-u.ac.jp/ hisaoi/ci_lab_e/research/pdf_file/multiobjective/EMO_ReSubmission_Ishibuchi-.html, April 2003.

[153] Ishibuchi, H., and Narukawa, K. Some issues on the implementation of local search in evolutionary multiobjective optimization. In *Genetic and Evolutionary Computation - GECCO 2004* (2004), vol. Volume 3102/2004 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 1246–1258.

[154] Ishibuchi, H., Yoshida, T., and Murata, T. Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. *Evolutionary Computation, IEEE Transactions on 7*, 2 (april 2003), 204 – 223.

[155] Iwamatsu, M. Co-evolutionary global optimization algorithm. In *CEC '02: Proceedings of the 2002 Congress on Evolutionary Computation* (Washington, DC, USA, 2002), vol. 2, IEEE Computer Society, pp. 1180–1184.

[156] Jaszkiewicz, A. Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research 137*, 1 (February 2002), 50–71.

[157] Jenkins, L. A bicriteria knapsack program for planning remediation of contaminated lightstation sites. *European Journal of Operational Research 140*, 2 (2002), 427 – 433.

[158] Joines, J., and Houck, C. On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In *Proceedings*

of the first IEEE Conference on Evolutionary Computation (Orlando, Florida, 1994), D. Fogel, Ed., IEEE Press, pp. 579–584.

[159] Kakimura, N., Makino, K., and Seimi, K. Computing knapsack solutions with cardinality robustness. In *Algorithms and Computation - 22nd International Symposium, ISAAC 2011* (December 2011), T. Asano, S.-I. Nakano, Y. Okamoto, and O. Watanabe, Eds., vol. 7074 of *Lecture Notes in Computer Science*, Springer, pp. 693–702.

[160] Kazarlis, S., and Petridis, V. Varying Fitness Functions in Genetic Algorithms: Studying the Rate of Increase of the Dynamic Penalty Terms. In *Proceedings of the 5th Parallel Problem Solving from Nature (PPSN V)* (Heidelberg, Germany, September 1998), A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds., Amsterdan, The Netherlands, Springer-Verlag, pp. 211–220.

[161] Kennedy, J., and Eberhart, R. Particle swarm optimization. In *Proc. IEEE International Conference on Neural Networks* (Perth, November 1995), vol. 4, pp. 1942–1948.

[162] Kim, J., and Myung, H. Hybrid Evolutionary Optimization Algorithm for Constrained Problems. In *Evolutionary Computation, Theory and Applications*, X. Yao, Ed. World Scientific, Singapore, 1999, pp. 256–295.

[163] Kim, J.-H., and Myung, H. Evolutionary programming techniques for constrained optimization problems. *IEEE Transactions on Evolutionary Computation 1* (July 1997), 129–140.

[164] Klamroth, K., and Wiecek, M. M. Dynamic programming approaches to the multiple criteria knapsack problem. *Naval Research Logistics 47*, 1 (2000), 57–76.

[165] Knowles, J. The multiobjective quadratic assignment problem (mqap). In web page accessed on December 2011 and available on http://dbkgroup.org/knowles/mQAP/, July 2003.

[166] Knowles, J., and Corne, D. On Metrics for Comparing Nondominated Sets. In *Congress on Evolutionary Computation (CEC'2002)* (Piscataway, New Jersey, May 2002), vol. 1, IEEE Service Center, pp. 711–716.

[167] Knowles, J., and Corne, D. Towards Landscape Analyses to Inform the Design of Hybrid Local Search for the Multiobjective Quadratic Assignment Problem. In *Soft Computing Systems: Design, Management and Applications* (Amsterdam, 2002), A. Abraham, J. R. del Solar, and M. Koppen, Eds., IOS Press, pp. 271–279. ISBN 1-58603-297-6.

[168] Knowles, J. D., and Corne, D. Instance generators and test suites for the multiobjective quadratic assignment problem. In *EMO* (2003), C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, Eds., vol. 2632 of *Lecture Notes in Computer Science*, Springer, pp. 295–310.

[169] Knowles, J. D., and Corne, D. W. The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimisation. In *1999 Congress on Evolutionary Computation* (Washington, D.C., July 1999), IEEE Service Center, pp. 98–105.

[170] Knowles, J. D., and Corne, D. W. Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy. *Evolutionary Computation 8*, 2 (2000), 149–172.

[171] Knuth, D. E. *The art of computer programming, volume 3: (2nd ed.) sorting and searching*. Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, USA, 1998.

[172] Kolman, B., and Beck, R. E. *Elementary Linear Programming with Applications, Second Edition (Computer Science and Scientific Computing)*, 2 ed. Academic Press, July 1995.

[173] Koster, A. M. C. A. *Frequency Assignment - Models and Algorithms*. PhD thesis, Maastricht University, 1999.

[174] Kostreva, M. M., Ogryczak, W., and Tonkyn, D. Relocation problems arising in conservation biology. *Computers and Mathematics with Applications 37*, 4-5 (1999), 135–150.

[175] Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)*, 1 ed. The MIT Press, December 1992.

[176] Koziel, S., and Michalewicz, Z. A decoder-based evolutionary algorithm for constrained parameter optimization problems. In *Proceedings of the 5th*

International Conference on Parallel Problem Solving from Nature (London, UK, 1998), PPSN V, Springer-Verlag, pp. 231–240.

[177] Koziel, S., and Michalewicz, Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evolutionary Computation 7* (March 1999), 19–44.

[178] Kreher, D. L. *Combinatorial Algorithms: Generation, Enumeration, and Search*. CRC, December 1998.

[179] Krink, T., and Thomsen, R. Self-organized criticality and mass extinction in evolutionary algorithms. In *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on* (2001), vol. 2, pp. 1155 –1161 vol. 2.

[180] Kuri-Morales, A., and Villegas-Quezada, C. A universal eclectic genetic algorithm for constrained optimization. In *Proceedings 6th European Congress on Intelligent Techniques & Soft Computing, EUFIT'98* (Mexico, 1998), pp. 518–522.

[181] Landt, J. The history of rfid. *Ieee Potentials 24*, 4 (2005), 8–11.

[182] Le, T. V. A Fuzzy Evolutionary Approach to Constrained Optimization Problems. In *Proceedings of the Second IEEE Conference on Evolutionary Computation* (Perth, November 1995), IEEE, pp. 274–278.

[183] Le Riche, R., and Haftka, R. T. Optimization of laminate stacking sequence for buckling load maximization by genetic algorithm. *AIAA Journal 31* (May 1993), 951–956.

[184] Lee, J. *A first course in combinatorial optimization*. Cambridge University Press, 2004.

[185] Leguizamón, G., and Coello, C. A. C. Boundary Search for Constrained Numerical Optimization Problems with an Algorithm Inspired on the Ant Colony Metaphor. *IEEE Transactions on Evolutionary Computation 13*, 2 (April 2009), 350–368.

[186] Leguizamón, G., and Coello-Coello, C. A Boundary Search based ACO Algorithm Coupled with Stochastic Ranking. In *2007 IEEE Congress on Evolutionary Computation (CEC'2007)* (Singapore, September 2007), IEEE Press, pp. 165–172.

[187] Leguizamón, G., and Coello Coello, C. Boundary Search for Constrained Numerical Optimization Problems. In *Constraint-Handling in Evolutionary Computation*, E. Mezura-Montes, Ed. Springer. Studies in Computational Intelligence, Volume 198, Berlin, 2009, ch. 2, pp. 25–49. ISBN 978-3-642-00618-0.

[188] Levine, J., and Ducatelle, F. Ant colony optimisation and local search for bin packing and cutting stock problems. *Journal of the Operational Research Society 55*, 7 (July 2004), 705–716.

[189] Lewis, A., and Abramson, D. An evolutionary programming algorithm for multi-objective optimisation. *The 2003 Congress on Evolutionary Computation, 2003. CEC '03. 3* (December 2003), 1926–1932.

[190] Lewis, A., Abramson, D., and Peachey, T. An evolutionary programming algorithm for automatic engineering design. In *Parallel Processing and Applied Mathematics*, R. Wyrzykowski, J. Dongarra, M. Paprzycki, and J. Wasniewski, Eds., vol. 3019 of *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2004, pp. 586–594. 10.1007/978-3-540-24669-5-77.

[191] Lewis, A., Mostaghim, S., and Randal, M. Evolutionary population dynamics and multi-objective optimisation problems. In *Multi-Objective Optimization in Computational Intelligence: Theory and Practice*, L. Bui and S. Alam, Eds. Information Science Reference, 2008, ch. 7.

[192] Lewis, A., Randall, M., Galehdar, A., Thiel, D., and Weis, G. Using ant colony optimisation to construct meander-line rfid antennas. In *Biologically-Inspired Optimisation Methods*, A. Lewis, S. Mostaghim, and M. Randall, Eds., vol. 210 of *Studies in Computational Intelligence*. Springer Berlin / Heidelberg, 2009, pp. 189–217.

[193] Lewis, A., Weis, G., Randall, M., Galehdar, A., and Thiel, D. Optimising efficiency and gain of small meander line rfid antennas using ant colony system. In *CEC'09: Proceedings of the Eleventh conference on Congress on Evolutionary Computation* (Piscataway, NJ, USA, May 2009), IEEE Press, pp. 1486–1492.

[194] Liepins, G. E., and Potter, W. D. *A Genetic Algorithm Approach to Multiple-Fault Diagnosis*. Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, New York, 1991, ch. 17, pp. 237–250.

[195] Liepins, G. E., and Vose, M. D. Representational issues in genetic optimization. *Journal of Experimental & Theoretical Artificial Intelligence 2*, 2 (1990), 101–115.

[196] Lorena, L. A. N., Narciso, M. G., and Beasley, J. E. A constructive genetic algorithm for the generalized assignment problem. Research Paper, 2002.

[197] Lu, Y.-Z., Chen, M.-R., and Chen, Y.-W. Studies on extremal optimization and its applications in solving realworld optimization problems. In *FOCI '07: Proceedings of the IEEE Symposium on Foundations of Computational Intelligence* (April 2007), IEEE, pp. 162–168.

[198] Luger, G. F. *Artificial Intelligence: Structures and Strategies for Complex Problem Solving*, 6th edition ed. Addison Wesley, USA, 2008.

[199] Luke, S. *Essentials of Metaheuristics*. Lulu, 2009.

[200] Malkevitch, J., and Lesser, L. M. *For All Practical Purposes: Mathematical Literacy in Today's World*, 8th ed. W. H. Freeman and Company, New York, NY, USA, 2009, ch. 3: Planning and Scheduling, pp. 67–102.

[201] Marrocco, G. Gain-optimized self-resonant meander line antennas for rfid applications. *Antennas and Wireless Propagation Letters, IEEE 2*, 1 (2003), 302 –305.

[202] Martello, S., and Toth, P. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., New York, NY, USA, 1990.

[203] Martello, S., and Toth, P. Lower bounds and reduction procedures for the bin packing problem. *Discrete Applied Mathematics 28*, 1 (July 1990), 59–70.

[204] Mathew, F., and Rego, C. Recent advances in heuristics for the capacitated minimum spanning tree problem. *The Decision Sciences Institute Conference (DSI)* (November 2006), 31021–31026.

[205] Menaï, M. E., and Batouche, M. Efficient initial solution to extremal optimization algorithm for weighted maxsat problem. In *IEA/AIE '2003: Proceedings of the 16th international conference on Developments in applied artificial intelligence* (2003), Springer Verlag Inc., pp. 592–603.

[206] Menaï, M. E., and Batouche, M. An effective heuristic algorithm for the maximum satisfiability problem. *Applied Intelligence 24*, 3 (June 2006), 227–239.

[207] Meshoul, S., and Batouche, M. Ant colony system with extremal dynamics for point matching and pose estimation. In *ICPR '02: Proceedings of the 16th International Conference on Pattern Recognition* (Washington, DC, USA, 2002), vol. 3, IEEE Computer Society, p. 30823.

[208] Michalewicz, Z. *Genetic algorithms + data structures = evolution programs.* Springer, 1992.

[209] Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*, third ed. Springer-Verlag, 1996.

[210] Michalewicz, Z., Dasgupta, D., Le Riche, R. G., and Schoenauer, M. Evolutionary algorithms for constrained engineering problems. *Comput. Ind. Eng. 30* (September 1996), 851–870.

[211] Michalewicz, Z., and Nazhiyath, G. Genocop iii: A co-evolutionary algorithm for numerical optimization problems with nonlinear constraints. In *Proceedings of IEEE International Conference on Evolutionary Computation* (November 1995), vol. 2, IEEE Press, pp. 647–651.

[212] Michalewicz, Z., and Xiao, J. Evaluation of paths in evolutionary planner/navigator. In *Proceedings of the 1995 International Workshop on Biologically Inspired Evolutionary Systems* (Tokyo, Japan, May 1995), pp. 45–52.

[213] Middleton, A. A. Improved extremal optimization for the Ising spin glass. *Physical Review E 69*, 5 (May 2004), 055701.

[214] Misevicius, A., Blazauskas, T., Blonskis, J., and Smolinskas, J. An overview of some heuristic algorithms for combinatorial optimization problems. *Information technology and control 1* (2004), 21–31.

[215] Montgomery, J. *Solution biases and pheromone representation selection in ant colony optimisation.* PhD thesis, School of Information Technology, Bond University, 2005.

[216] Montgomery, J., Randall, M., and Lewis, A. Differential evolution for rfid antenna design: a comparison with ant colony optimisation. In *In Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference, GECCO 2011* (Dublin, Ireland, July 2011), N. Krasnogor and P. L. Lanzi, Eds., ACM, pp. 673–680.

[217] Moscato, P. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. Tech. Rep. Caltech Concurrent Computation Program, Report. 826, California Institute of Technology, Pasadena, California, USA, 1989.

[218] Moscato, P., and Cotta, C. A gentle introduction to memetic algorithms. *Handbook of Metaheuristics 57* (2003), 105–144.

[219] Moscato, P., and Cotta, C. *Handbook of Approximation Algorithms and Metaheuristics*. Chapman and Hall/CRC Computer and Information Science Series. CRC Press, May 2007, ch. Memetic Algorithms, pp. 27–1 – 27–12.

[220] Moscato, P., and Cotta, C. A modern introduction to memetic algorithms. In *Handbook of Metaheuristics*, M. Gendreau and J.-Y. Potvin, Eds., vol. 146 of *International Series in Operations Research, Management Science*. Springer US, 2010, pp. 141–183.

[221] Moscato, P., and Cotta Porras, C. Una introducción a los algoritmos memeticos. *Inteligencia Artificial, Revista Iberoamericana de IA 7*, 19 (2003), 131–148.

[222] Moser, I., and Hendtlass, T. On the behaviour of extremal optimisation when solving problems with hidden dynamics. In *IEA/AIE '06: The 19th International Conference on Industrial, Engineering & Other Applications of Applied Intelligent Systems* (2006), M. Ali and R. Dapoigny, Eds., vol. 4031 of *Lecture Notes in Computer Science*, Springer, pp. 292–301.

[223] Moser, I., and Hendtlass, T. Solving problems with hidden dynamics : comparison of extremal optimisation and ant colony system. In *IEEE Congress on Evolutionary Computation, 2006. CEC 2006.* (September 2006), IEEE, pp. 1248–1255.

[224] Moser, I., and Hendtlass, T. Solving dynamic single-runway aircraft landing problems with extremal optimisation. In *IEEE Symposium on Computational Intelligence in Scheduling, 2007.* (April 2007), IEEE, pp. 206 – 211.

[225] Mühlenbein, H. Parallel genetic algorithms in combinatorial optimization. *Computer science and operations research* (1992), 441–456.

[226] Musser, D. R. Introspective sorting and selection algorithms. *Softw. Pract. Exper. 27* (August 1997), 983–993.

[227] Néda, Z., Florian, R., Ravasz, M., Libál, A., and Györgyi, G. Phase transition in an optimal clusterization model. *Physica A: Statistical Mechanics and its Applications 362*, 2 (2006), 357 – 368.

[228] Neebe, A. W., and Rao, M. R. An algorithm for the fixed-charge assigning users to sources problem. *Journal of the Operational Research Society 34*, 11 (11 1983), 1107–1113.

[229] Nocedal, J., and Wright, S. J. *Numerical Optimization*, 2nd ed. Springer Series in Operations Research. Springer, New York, July 2006.

[230] Norman, B. A., and Smith, A. E. Random Keys Genetic Algorithm with Adaptive Penalty Function for Optimization of Constrained Facility Layout Problems. In *Proceedings of the 1997 International Conference on Evolutionary Computation* (Indianapolis, Indiana, 1997), T. Bäck, Z. Michalewicz, and X. Yao, Eds., IEEE, pp. 407–411.

[231] Oberdorf, R., Ferguson, A., Jacobsen, J. L., and Kondev, J. Secondary structures in long compact polymers. *Phys. Rev. E 74* (Nov 2006), 051801.

[232] Onody, R. N., and de Castro, P. A. Optimization and self-organized criticality in a magnetic system. *Physica A: Statistical Mechanics and its Applications 322* (2003), 247 – 255.

[233] Orvosh, D., and Davis, L. Using a genetic algorithm to optimize problems with feasability constraints. In *International Conference on Evolutionary Computation* (1994), pp. 548–553.

[234] Osman, I. H., and Kelly, J. P., Eds. *Meta-Heuristics: Theory and Applications*. Kluwer Academic Publishers, Norwell, MA, USA, 1996.

[235] Palmer, C. C., and Kershenbaum, A. Representing Trees in Genetic Algorithms. In *Proceedings of the First IEEE Conference on Evolutionary Computation* (Piscataway, New Jersey, 1994), Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, Eds., IEEE Press, pp. 379–384.

[236] Paredis, J. Co-evolutionary constraint satisfaction. In *PPSN III: Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel Problem Solving from Nature* (London, UK, 1994), vol. 866 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 46–55.

[237] Paredis, J. Towards balanced coevolution. In *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature (PPSN VI)* (London, UK, 2000), Springer-Verlag, pp. 497–506.

[238] Petersen, C. C. Computational experience with variants of the balas algorithm applied to the selection of r&d projects. *MANAGEMENT SCIENCE 13*, 9 (1967), 736–750.

[239] Powell, D., and Skolnick, M. M. Using genetic algorithms in engineering design optimization with non-linear constraints. In *Proceedings of the 5th International Conference on Genetic Algorithms* (San Francisco, CA, USA, 1993), Morgan Kaufmann Publishers Inc., pp. 424–431.

[240] Puchinger, J., Raidl, G. R., and Pferschy, U. The core concept for the multidimensional knapsack problem. *Science 3906/2006* (2006), 195–208.

[241] Raidl, G. An improved genetic algorithm for the multiconstrained 0-1 knapsack problem. In *Proceedings of the 5th IEEE International Conference on Evolutionary Computation* (May 1998), IEEE Press, pp. 207–211.

[242] Randall, M. *Algorithms: Machines of the Mind*, 1st ed. ed. Bond University Press, Gold Coast, QLD, Australia, 2004.

[243] Randall, M. Enhancements to extremal optimisation for generalised assignment. In *ACAL* (November 2007), M. Randall, H. A. Abbass, and J. Wiles, Eds., vol. 4828 of *Lecture Notes in Artificial Intelligence*, Springer, pp. 369–380.

[244] Randall, M., Hendtlass, T., and Lewis, A. Extremal optimisation for assignment type problems. In *Biologically-Inspired Optimisation Methods: Parallel Algorithms, Systems and Applications*, A. Lewis, S. Mostaghim, and M. Randall, Eds., vol. 210 of *Studies in Computational Intelligence*. Springer-Verlag, May 2009, pp. 139–164.

[245] Randall, M., and Lewis, A. An extended extremal optimisation model for parallel architectures. In *E-SCIENCE '06: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing* (Washington, December 2006), Biologically-inspired Optimisation Methods for Parallel and Distributed Architectures: Algorithms, Systems and Applications, IEEE Computer Society, p. 114.

[246] Randall, M., Lewis, A., Galehdar, A., and Thiel, D. Using ant colony optimisation to improve the efficiency of small meander line rfid antennas. In *Third International Conference on e-Science and Grid Computing, e-Science 2007* (Bangalore, India, December 2007), IEEE Computer Society, pp. 345 – 351.

[247] Reeves, C. R., Ed. *Modern heuristic techniques for combinatorial problems*. John Wiley & Sons, Inc., New York, NY, USA, 1993.

[248] Reinelt, G. *The traveling salesman: computational solutions for TSP applications*. Springer-Verlag, Berlin, Heidelberg, 1994.

[249] Reynolds, R. G., Michalewicz, Z., and Cavaretta, M. Using cultural algorithms for constraint handling in GENOCOP. In *Proceedings of the Fourth Annual Conference on Evolutionary Programming*, J. R. McDonnell, R. G. Reynolds, and D. B. Fogel, Eds. MIT Press, Cambridge, Massachusetts, 1995, pp. 298–305.

[250] Riche, R. G. L., and Haftka, R. T. Optimization of Laminate Stacking Sequence for Buckling Load Maximization by Genetic Algorithm. *AIAA Journal 31*, 5 (1993), 951–970.

[251] Riche, R. G. L., Knopf-Lenoir, C., and Haftka, R. T. A Segregated Genetic Algorithm for Constrained Structural Optimization. In *Proceedings of the Sixth International Conference on Genetic Algorithms (ICGA-95)* (San Mateo, California, July 1995), L. J. Eshelman, Ed., University of Pittsburgh, Morgan Kaufmann Publishers, pp. 558–565.

[252] Runarsson, T. P., and Yao, X. Search biases in constrained evolutionary optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part C 35*, 2 (2005), 233–243.

[253] Sahni, S., and Gonzalez, T. P-complete approximation problems. *J. ACM 23* (July 1976), 555–565.

[254] Schaffer, J. D. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, 1984.

[255] Schaffer, J. D. Multiple Objective Optimization with Vector Evaluated Genetic Algorithms. In *Genetic Algorithms and their Applications: Proceedings*

of the First International Conference on Genetic Algorithms (1985), Lawrence Erlbaum, pp. 93–100.

[256] Schaffer, J. D., and Grefenstette, J. J. Multiobjective Learning via Genetic Algorithms. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence (IJCAI-85)* (Los Angeles, California, 1985), AAAI, pp. 593–595.

[257] Schoenauer, M., and Xanthakis, S. Constrained ga optimization. In *Proceedings of the 5th International Conference on Genetic Algorithms* (Urbana-Champaign, IL, USA, June 1993), S. Forrest, Ed., Morgan Kaufmann, pp. 573–580.

[258] Schott, J. R. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[259] Shadbolt, N. Nature-inspired computing. *IEEE Intelligent Systems 19* (2004), 2–3.

[260] Shannon, C. A., and McKinney, D. An evolutionary algorithm for assigning students to courses. In *FLAIRS Conference* (2011), R. C. Murray and P. M. McCarthy, Eds., AAAI Press.

[261] Shmygelska, A. An extremal optimization search method for the protein folding problem: The go-model example. In *GECCO '07: Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation* (New York, NY, USA, 2007), D. Thierens, Ed., ACM, pp. 2572–2579.

[262] Shraideh, A., Camus, H., and Yim., P. New generalized assignment problem with identified first-used bin. In *In: VI ALIO/EURO Workshop on Applied Combinatorial Optimization* (Universidad de Buenos Aires, December 2008), Facultad de Ciencias Exactas y Naturales.

[263] Siedlecki, W., and Sklanski, J. Constrained Genetic Optimization via Dynamic Reward-Penalty Balancing and Its Use in Pattern Recognition. In *Proceedings of the Third International Conference on Genetic Algorithms (ICGA-89)* (San Mateo, California, June 1989), J. D. Schaffer, Ed., George Mason University, Morgan Kaufmann Publishers, pp. 141–150.

[264] Sivarajan, K., McEliece, R., and Ketchum, J. Channel assignment in cellular radio. In *Vehicular Technology Conference, 1989, IEEE 39th* (May 1989), vol. 2, pp. 846 –850.

[265] Smith, A. E., and Coit, D. W. Penalty functions. In *Handbook of Evolutionary Computation*, T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds. Institute of Physics Publishing and Oxford University Press, Bristol, New York, 1997, pp. C5.2:1–6.

[266] Smith, A. E., and Tate, D. M. Genetic Optimization Using a Penalty Function. In *Proceedings of the Fifth International Conference on Genetic Algorithms (ICGA-93)* (San Mateo, California, July 1993), S. Forrest, Ed., University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers, pp. 499–503.

[267] Sokal, A. D. Monte Carlo methods for the selfavoiding walk. *Monte Carlo and Molecular Dynamics Simulations in Polymer Science* (1994). Published in Monte Carlo and Molecular Dynamics Simulations in Polymer Science, edited by Kurt Binder, Oxford University Press, 1995, pp. 47-124.

[268] Sousa, F. L., Ramos, F. M., Galski, R. L., and Muraoka, I. Generalized extremal optimization: A new meta-heuristic inspired by a model of natural evolution. In *Recent Developments in Biologically Inspired Computing*, L. N. Castro and F. J. Von Zuben, Eds. Idea Group, 2005, ch. 3, pp. 41–60.

[269] Sousa, F. L. D., and Ramos, F. M. Function optimization using extremal dynamics. In *Proceeding of 4th International Conference on Inverse Problems in Engineering* (Rio de Janeiro, Brazil, 2002).

[270] Sousa, F. L. D., Ramos, F. M., Paglione, P., and Girardi, R. M. New stochastic algorithm for design optimization. *AIAA Journal 41*, 9 (2003), 1808–1818.

[271] Sousa, F. L. D., Soeiro, F. J. C. P., Neto, A. J. S., and Ramos, F. M. Application of the generalized extremal optimization algorithm to an inverse radiative transfer problem. *Inverse Problems in Science and Engineering 15*, 7 (January 2007), 699–714.

[272] Sousa, F. L. d., Vlassov, V., and Ramos, F. M. Generalized extremal optimization for solving complex optimal design problems. In *GECCO '03: Genetic and Evolutionary Computation* (2003), vol. 2723 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 375–376.

[273] Sousa, F. L. D., Vlassov, V., and Ramos, F. M. Generalized extremal optimization: An application in heat pipe design. *Applied Mathematical Modelling 28*, 10 (October 2004), 911–931.

[274] Sousa, F. L. D., Vlassov, V., and Ramos, F. M. Heat pipe design through generalized extremal optimization. *Heat Transfer Engineering 25*, 7 (2004), 35–45.

[275] Srinivas, N., and Deb, K. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation 2*, 3 (Fall 1994), 221–248.

[276] Stockman, H. Communication by means of reflected power. *Proceedings of the IRE 36*, 10 (oct. 1948), 1196 – 1204.

[277] Storn, R., and Price, K. Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization 11*, 4 (1997), 341–359.

[278] Stützle, T., and Hoos, H. H. Max-min ant system. *Future Gener. Comput. Syst. 16* (June 2000), 889–914.

[279] Sungur, I., Ordonez, F., and Dessouky, M. A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty. *IIE Transactions 40*, 5 (2008), 509–523.

[280] Suppapitnarm, A., Seffen, K., Parks, G., Clarkson, P., , and and. Design by multiobjective optimisation using simulated annealing. In *The 12th International Conference on Engineering Design, ICED'99* (August 1999), vol. 3 of *WDK Workshop Design Konstruktion*, Technische Universitat Munchen, pp. 1395–1400.

[281] Surry, P. D., and Radcliffe, N. J. The COMOGA Method: Constrained Optimisation by Multiobjective Genetic Algorithms. *Control and Cybernetics 26*, 3 (1997), 391–412.

[282] Svenson, P. Extremal optimization for sensor report pre-processing. *SPIE proceedings series 5429* (2004), 162–171.

[283] Sywerda, G. Uniform crossover in genetic algorithms. In *Proceedings of the third international conference on Genetic algorithms* (San Francisco, CA, USA, 1989), Morgan Kaufmann Publishers Inc., pp. 2–9.

[284] Tate, D. M., and Smith, A. E. Expected allele coverage and the role of mutation in genetic algorithms. In *Proceedings of the 5th International Conference on Genetic Algorithms* (1993), S. Forrest, Ed., Morgan Kaufmann, pp. 31–37.

[285] Tavares, J., Pereira, F., and Costa, E. Multidimensional knapsack problem: The influence of representation. Tech. Rep. TR 2007/003, Centre for Informatics and Systems of the University of Coimbra (CISUC), February 2007.

[286] Toscano Pulido, G., and Coello Coello, C. A. The Micro Genetic Algorithm 2: Towards Online Adaptation in Evolutionary Multiobjective Optimization. In *Evolutionary Multi-Criterion Optimization. Second International Conference, EMO 2003* (Faro, Portugal, April 2003), C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele, Eds., Springer. Lecture Notes in Computer Science. Volume 2632, pp. 252–266.

[287] Uyar, S., and Eryiğit, G. Improvements to penalty-based evolutionary algorithms for the multi-dimensional knapsack problem using a gene-based adaptive mutation approach. In *GECCO '05: Proceedings of the 2005 conference on Genetic and evolutionary computation* (New York, NY, USA, 2005), ACM, pp. 1257–1264.

[288] Vasquez, M., and Hao, J.-K. A hybrid approach for the 0-1 multidimensional knapsack problem. In *Proceedings of International Joint Conerence Artificial Intelligence* (Seatle, August 2001), pp. 328–333.

[289] Vasquez, M., and Vimont, Y. Improved results on the 0-1 multidimensional knapsack problem. *European Journal of Operational Research 165*, 1 (August 2005), 70–81.

[290] Veldhuizen, D. A. V. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations.* PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[291] Veldhuizen, D. A. V., and Lamont, G. B. Evolutionary Computation and Convergence to a Pareto Front. In *Late Breaking Papers at the Genetic Programming 1998 Conference* (Stanford University, California, July 1998), J. R. Koza, Ed., Stanford University Bookstore, pp. 221–228.

[292] Veldhuizen, D. A. V., and Lamont, G. B. Genetic Algorithms, Building Blocks, and Multiobjective Optimization. In *Proceedings of the 1999 Genetic and*

Evolutionary Computation Conference. Workshop Program (Orlando, Florida, July 1999), A. S. Wu, Ed., pp. 125–126.

[293] Veldhuizen, D. A. V., and Lamont, G. B. Multiobjective Evolutionary Algorithm Test Suites. In *Proceedings of the 1999 ACM Symposium on Applied Computing* (San Antonio, Texas, 1999), J. Carroll, H. Haddad, D. Oppenheim, B. Bryant, and G. B. Lamont, Eds., ACM, pp. 351–357.

[294] Venkatraman, S., and Yen, G. A generic framework for constrained optimization using genetic algorithms. *Evolutionary Computation, IEEE Transactions on 9*, 4 (August 2005), 424 – 435.

[295] Vimont, Y., Boussier, S., and Vasquez, M. Reduced costs propagation in an efficient implicit enumeration for the 0-1 multidimensional knapsack problem. *Journal of Combinatorial Optimization 15*, 2 (2008), 165–178.

[296] Visée, M., Teghem, J., Pirlot, M., and Ulungu, E. L. Two-phases method and branch and bound procedures to solve the bi-objective knapsack problem. *Journal of Global Optimization 12* (1998), 139–155.

[297] Wang, J.-S. Transition matrix monte carlo and flat-histogram algorithm. *AIP Conference Proceedings 690*, 1 (2003), 344–348.

[298] Wang, J. S., and Okabe, Y. A comparison of extremal optimization with flat-histogram dynamics for finding spin-glass ground states. *Journal of the Physical Society of Japan 72* (2003), 1380.

[299] Webb, J. W., and Reis, R. A. *Programmable Logic Controllers: Principles and Applications*, 5th ed. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2002.

[300] Weis, G., Lewis, A., Randall, M., Galehdar, A., and Thiel, D. Local search for ant colony system to improve the efficiency of small meander line rfid antennas. In *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2008* (June 2008), IEEE, pp. 1708 –1713.

[301] Weis, G., Lewis, A., Randall, M., and Thiel, D. Pheromone pre-seeding for the construction of rfid antenna structures using aco. In *Sixth International Conference on e-Science, e-Science 2010* (Brisbane, QLD, Australia, December 2010), IEEE Computer Society, pp. 161–167.

[302] Whitley, D. The genitor algorithm and selection pressure: why rank-based allocation of reproductive trials is best. In *Proceedings of the third international conference on Genetic algorithms* (San Francisco, CA, USA, 1989), Morgan Kaufmann Publishers Inc., pp. 116–121.

[303] Williams, J. W. J. Algorithm 232: Heapsort. *Communication of the ACM 7* (1964), 347–348.

[304] Woeginger, G. J. Exact algorithms for np-hard problems: a survey. *Combinatorial Optimization - Eureka, You Shrink!* (2003), 185–207.

[305] Wolpert, D. H., and Macready, W. G. No free lunch theorems for optimization. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION 1*, 1 (1997), 67–82.

[306] Wolsey, L. A. *Integer Programming*, 1st ed. Interscience Series in Discrete Mathematics and Optimization. Wiley, December 1998.

[307] Xiao, J., Michalewicz, Z., and Zhang, L. Evolutionary planner/navigator: operator performance and self-tuning. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on* (May 1996), pp. 366 –371.

[308] Xiao, J., Michalewicz, Z., Zhang, L., and Trojanowski, K. Adaptive Evolutionary Planner/Navigator for Mobile Robots. *IEEE Transactions on Evolutionary Computation 1*, 1 (April 1997), 18–28.

[309] Yang, X.-S. *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008.

[310] Yom-Tov, E., Grossman, A., and Inbar, G. F. Movement-related potentials during the performance of a motor task I: The effect of learning and force. *Biological Cybernetics 85*, 5 (2001), 395–399.

[311] Yoo, J., and Hajela, P. Enhanced GA Based Search Through Immune System Modeling. In *3rd World Congress on Structural and Multidisciplinary Optimization* (Niagara Falls, New York, May 1999).

[312] Young, A. *Spin glasses and random fields*. Directions in condensed matter physics. World Scientific, 1998.

[313] Zadeh, L. Optimality and non-scalar-valued performance criteria. *Automatic Control, IEEE Transactions on 8*, 1 (jan 1963), 59 – 60.

[314] Zhang, N., and Zeng, C. Reference energy extremal optimization: A stochastic search algorithm applied to computational protein design. *J. Comput. Chem. 29*, 11 (2008), 1762–1771.

[315] Zhou, T., Bai, W.-J., Cheng, L.-J., and Wang, B.-H. Continuous extremal optimization for Lennard-Jones clusters. *Physical Review E 72*, 1 (July 2005), 016702.

[316] Zhou, Y., Li, Y., He, J., and Kang, L. Multi-objective and mgg evolutionary algorithm for constrained optimization. In *Evolutionary Computation, 2003. CEC '03. The 2003 Congress on* (Canberra, Australia, December 2003), vol. 1, pp. 1–5.

[317] zhuo Huang, F., Wang, L., and He, Q. A hybrid differential evolution with double populations for constrained optimization. *IEEE Congress on Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence)* (June 2008), 18–25.

[318] Zitzler, E. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[319] Zitzler, E., Deb, K., and Thiele, L. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation 8*, 2 (Summer 2000), 173–195.

[320] Zitzler, E., and Laumanns, M. Test problem suite. In web page accessed on December 2011 and available on http://www.tik.ee.ethz.ch/sop/download/supplementary/testProblemSuite, June 2008.

[321] Zitzler, E., Laumanns, M., and Bleuler, S. A tutorial on evolutionary multiobjective optimization. In *Metaheuristics for Multiobjective Optimisation* (2004), X. Gandibleux, M. Sevaux, K. Sorensen, and V. T'Kindt, Eds., vol. 535 of *Lecture Notes in Economics and Mathematical Systems*, Springer, pp. 3–38.

[322] Zitzler, E., Laumanns, M., and Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In *EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems* (Athens, Greece, 2001), K. Giannakoglou, D. Tsahalis, J. Periaux, P. Papailou, and T. Fogarty, Eds., pp. 95–100.

[323] Zitzler, E., and Thiele, L. Multiobjective Optimization Using Evolutionary Algorithms—A Comparative Study. In *Parallel Problem Solving from Nature V* (Amsterdam, September 1998), A. E. Eiben, Ed., Springer-Verlag, pp. 292–301.

[324] Zitzler, E., and Thiele, L. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *Evolutionary Computation, IEEE Transactions on 3*, 4 (Nov 1999), 257–271.

[325] Zydallis, J. B. *Explicit Building-Block Multiobjective Genetic Algorithms: Theory, Analysis, and Development*. PhD thesis, Air Force Institute of Technology, Department of the Air Force, Air University, Wright-Patterson, Airforce Base, Ohio, USA, March 2003.

[326] Zydallis, J. B., Veldhuizen, D. A. V., and Lamont, G. B. A Statistical Comparison of Multiobjective Evolutionary Algorithms Including the MOMGA–II. In *First International Conference on Evolutionary Multi-Criterion Optimization*, E. Zitzler, K. Deb, L. Thiele, C. A. C. Coello, and D. Corne, Eds. Springer-Verlag. Lecture Notes in Computer Science No. 1993, 2001, pp. 226–240.